НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

На правах рукопису
УДК 519.2

До захисту допущено
В. о. завідувача кафедри ММСА
О.Л.Тимощук
«___» _____ 2019 р.

Магістерська дисертація

на здобуття ступеня магістра за спеціальністю 124 Системний аналіз
на тему: «Системна класифікація кардіосигналів для оцінювання стану здоров'я людини»

Виконала:
студентка ІІ курсу, групи КА-81мп
Мнухіна Катерина Олександрівна _____

Керівник:
професор кафедри ММСА,
д.т.н., проф. Данилов В.Я. _____

Рецензент:

_____

Засвідчую, що у цій магістерській дисертації
немає запозичень з праць інших авторів
без відповідних посилань

Студент _____

Київ
2019

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти — другий (магістерський)
Спеціальність — 124 «Системний аналіз»

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри ММСА
О. Л. Тимощук
«___» _____ 2019 р.

ЗАВДАННЯ
на магістерську дисертацію студентці Мнухіній Катерині Олександрівні

1. Тема дисертації: «Системна класифікація кардіосигналів для оцінювання стану здоров'я людини», науковий керівник дисертації Данилов Валерій Якович, д.т.н., професор, затверджені наказом по університету від «___» _____ № _____

2. Термін подання студентом дисертації: 13 грудня 2019 р.

3. Об’єкт дослідження: робота серця та її вираження в кардіосигналах

4. Предмет дослідження: методи математичного моделювання та оптимізації, які застосовуються в задачах класифікації.

5. Перелік завдань, які потрібно розробити:
1) дослідити сучасний стан та особливості застосування математичного моделювання та оптимізації у вирішенні проблеми розпізнавання;
2) розробити математичну модель за допомогою прихованої Марківської мережі;
3) розв’язати розроблену математичну модель та на її основі створити програмний продукт;
4) пошук даних для застосування в програмі;
5) розробити стартап-проект виведення на ринок результатів дослідження;

6) розробити концептуальні висновки за результатами наукового дослідження

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:

1) Графічні матеріали та таблиці теоритичної частини;

2) Приклади функціонування створеного програмного продукту (рис.);

3) Таблиці у розділі стартап-проекту.

7. Дата видачі завдання: 05 вересня 2019 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації
1.	Концептуальний вступ дисертації. Формулювання об’єкта, предмета, цілі, завдань, новизни, практичної значущості результатів	05.09.2019—19.09.2019
2.	Перший розділ. Огляд літературно-інформаційних джерел. Понятійно-категоріальний апарат. Характеристика об’єкта	20.09.2019—27.09.2019
3.	Другий розділ. Розробка математичної моделі для задачі розпізнавання емоцій людини за мовленням, її розв’язок	27.09.2019—15.10.2019
4.	Третій розділ. Метод побудови прихованої марківської мережі. Пошук даних	15.10.2019—05.11.2019
5.	Четвертий розділ. Стартап-проект	05.11.2019—15.11.2019
6.	Концептуальні висновки. Перспективи розвитку отриманих рішень	15.11.2019—21.11.2019

Студент К.О. Мнухіна

Науковий керівник дисертації В.Я. Данилов

РЕФЕРАТ

Магістерська дисертація: 93 с., 26 рис., 27 табл, 19 джерел.

Мета роботи – розробка покращеної системи автоматичної класифікації форм ЕКГ, заснованого на використанні штучних нейронних мереж.

Об'єктом дослідження є автоматична класифікації форм електрокардіограм.

Предметом дослідження є сучасні алгоритми та методи класифікації форм ЕКГ.

В роботі досліджується проблема розпізнавання стану здоров’я людини, шляхом аналізу її електрокардіограми, а також програмні засоби для реалізації поставленої задачі.

Виконано аналіз методів обробки цифрових сигналів, проведено дослідження, аналіз математичних структур та сучасних методів класифікації.

Результатом роботи є аналіз ефективності сучасних підходів до вирішення задачі класифікації кардіограм, розробка власного модифікованого алгоритму зі збільшенням використаних ознак та допоміжним визначенням певних показників. Під час роботи реалізовано програмну реалізацію запропонованого алгоритму на мовах програмування Python та R, з використанням різних бібліотек для аналізу даних.

Програмний продукт реалізовано з використанням мов програмування Python та R, що надають широкий спектр бібліотек для обробки та аналізу даних.

КЛАСИФІКАЦІЯ, ЕЛЕКТРОКАРДІОГРАМА, НЕЙРОННІ МЕРЕЖІ, QRS-КОМПЛЕКС, PYTHON

ABSTRACT

Master's Thesis: 93 pages., 26 images., 27 tables, 19 sources.

The purpose of the work is to develop an automatic system for the ECG classification based on artificial neural networks.

The paper examines the problem of human health control by analyzing its electrocardiogram, and develop a software tools for the task.

The analysis of processing digital signals is carried out, analysis of mathematical structures and modern methods of classification are carried out, the mathematical results were based on the signals recognition and modern methods of classification. The result of the work is the analysis of the effectiveness of modern approaches to solving the problem of ECG classification, own modified algorithm development with increase of used signs and auxiliary definition of certain indicators. The software implemented in Python and R programming languages while using different libraries for data analysis.

The software is implemented using Python and R programming languages, which provide a wide range of libraries for data processing and analysis.

CLASSIFICATION, CARDIOGRAPHY, NEURAL NETWORKS, QRS COMPLEX, PYTHON

ЗМІСТ

ВСТУП

Актуальність роботи. Застосування методів штучного інтелекту (ШІ) стало надзвичайно важливим напрямком в електрокардіографії (ЕКГ) для розпізнавання різних типів хвороб, у тому числі аритмій. Під аритмією як правило розуміється будь-якого роду порушення в регулярної ритмічної активності серця (нехарактерні зміни в амплітуді ритму, його тривалості і формі). З діагностичної точки зору, найкраща інформація про присутність/відсутність аритмії міститься в QRS-комплексі - гострій дво- або трифазній хвилі амплітудою близько 1 мВ і триваліcтю приблизно від 80 до 100 мс.

На даний момент існує досить багато рішень для розробки системи автоматичного аналізу та розпізнавання форм ЕКГ в реальному масштабі часу [1-6]. Залежно від обраного способу обробки сигналу і його реалізації можна виділити статистичні та синтаксичні методи [7]. На сьогоднішній день реалізація моделі апріорної оцінки результатів з використанням методів ШІ, особливо нейронних мереж, стала дуже важливим підходом. Найбільш відомими є: техніка багатошарового перцептрона (БП) [2], карти Кохонена [1, 3], метод опорних векторів (МОВ) [5], квантизация векторів при навчанні (КВН) [1], лінійні дискримінантні системи (ЛДС) [6], нечіткі та нейро-нечіткі системи [8], а також комбінації різних підходів, заснованих на нейронних мережах, так звані «гібридні системи» [4].

Класична система розпізнавання серцевих скорочень, заснована на нейронних мережах, зазвичай навчає різні моделі, використовуючи різноманітні структури мереж або методи попередньої обробки даних, потім вибирає найкращий, а решту відкидає. Однак кожен метод обробки даних може бути чутливий до артефактів і викидів. Тому необхідно об’єднати наявну

інформацію в одну кінцеву систему розпізнавання образів, яка створить класифікатор високої якості з мінімальною помилкою класифікації.

Мета роботи –розробка системи автоматичної класифікації форм ЕКГ, заснованого на використанні штучних нейронних мереж. Для досягнення поставленої мети в роботі вирішуються наступні завдання:

- ознайомлення з існуючими методами класифікації форм ЕКГ;
- розробка алгоритму класифікації форм ЕКГ на основі штучних нейронних мереж;
- тестування розробленого алгоритму;
- застосування розробленого і протестованого алгоритму для аналізу і класифікації сигналів ЕКГ, отриманих при діагностиці стану реальних пацієнтів.

Результати роботи показали ефективність застосування класифікатора ЕКГ на основі штучних нейронних мереж. За допомогою розробленого алгоритму з'явилася можливість розділяти різні типи форм ЕКГ з використанням мінімального набору характеристик сигналу

Наукова і практична цінність роботи визначаються можливістю використання розробленого і програмно-реалізованого алгоритму класифікації форм ЕКГ для швидкої і точної класифікації великої кількості даних з метою своєчасної постановки діагнозу і надання необхідної допомоги пацієнту.

РОЗДІЛ 1 ЗАГАЛЬНІ ВІДОМОСТІ І ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 Основні відомості про електрокардіографію

В людському організмі серце є найбільш важливим з органів, без його правильної роботи повноцінне функціонування всіх інших систем є неможливим. Хвороби, пов'язані з порушеннями роботи серцево-судинної системи, є найпоширенішими на сьогоднішній день. Їх частота суттєво зросла за останні десятиліття, у тому числі як причини смертності населення. Причиною цього може бути зниження фізичної активності, підвищення рівня стресу у людей і загального навантаження на нервову систему, шкідливий вплив виробничих факторів і т. д. У зв'язку із цим кожна людина має час від часу проходити дослідження серця і серцево-судинної системи.

На щастя, в сучасній кардіології існує велика кількість діагностичних методів, що дозволяють вчасно виявляти відхилення в роботі серця і визначати рівень їх загрози. Серед методів діагностики можна виділити ЕКГ картування, холтерівське моніторування, УЗД серця, фонокардіографію та інші методи, однак найпоширенішим є метод електрокардіографії (ЕКГ).

Електрокардіографія що вивчає електричні сигнали, що виникають у міокарді серця. Графічне зображення електричних потенціалів серця називають електрокардіограмою (ЕКГ). Найважливішою перевагою методу ЕКГ є те, що форма хвиль кардіограми змінюється при наявності серцево-судинних захворювань і різних патологій, що дозволяє робити попередні висновки навіть при візуальному аналізі.

За допомогою двох електродів, що приєднуються до електрокардіографа, з поверхні тіла пацієнта фіксують значення електричних потенціалів. Загальноприйнятими та обов'язковими в електрокардіографії є 12 відведень.

Двополюсні від кінцівок, або стандартні відведення Ейнтховена (рисунок 1.1).

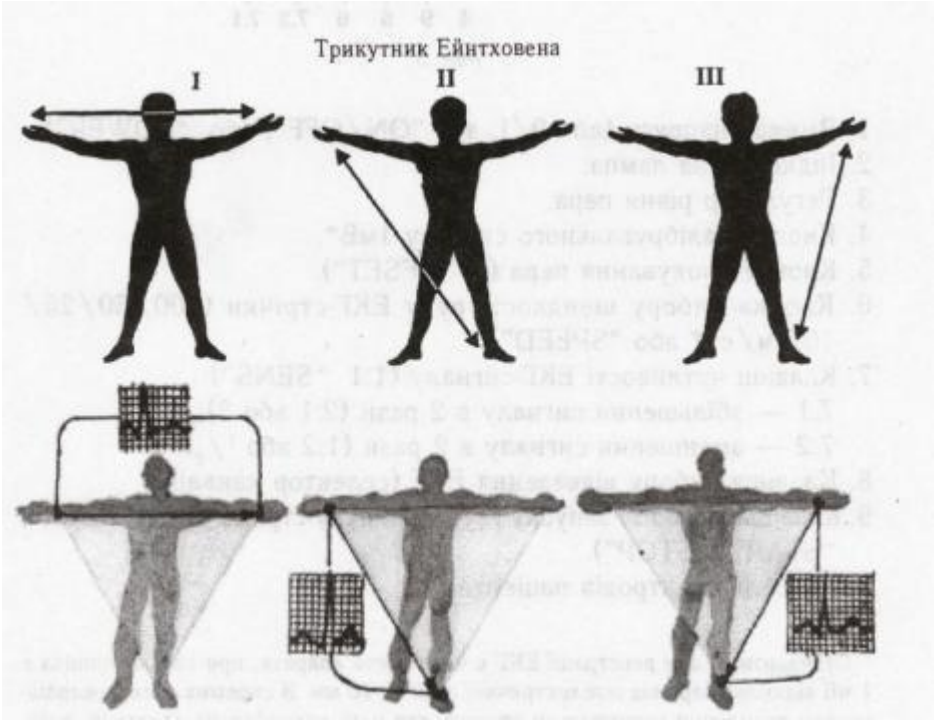

Рисунок 1.1 – Відведення Ейнтховена

Однополюсні підсилені, або відведення Гольдбергера (рисунок 1.2).

- I — між правою і лівою руками (червоний – жовтий);
- II — між правою рукою і лівою ногою (червоний – зелений);
- III — між лівою рукою і лівою ногою (жовтий – зелений).

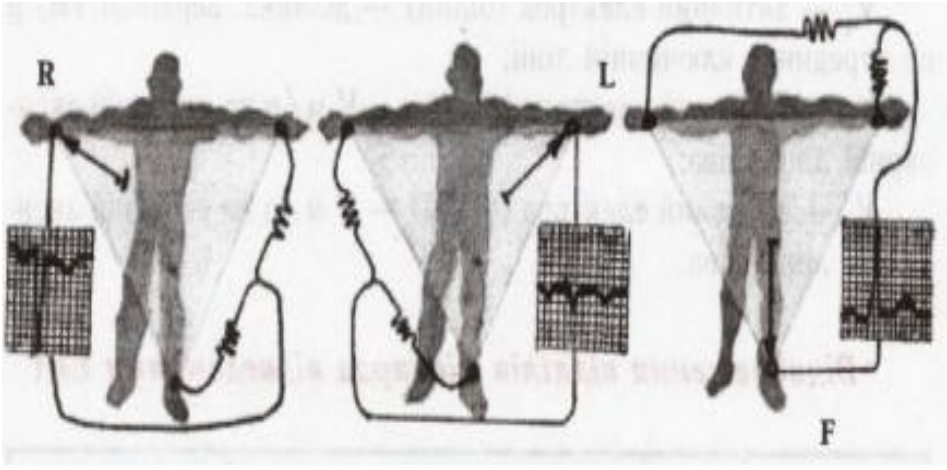

Рисунок 1.2 -- Відведення Гольдбергера

Грудні відведення Вільсона (рисунок 1.3).

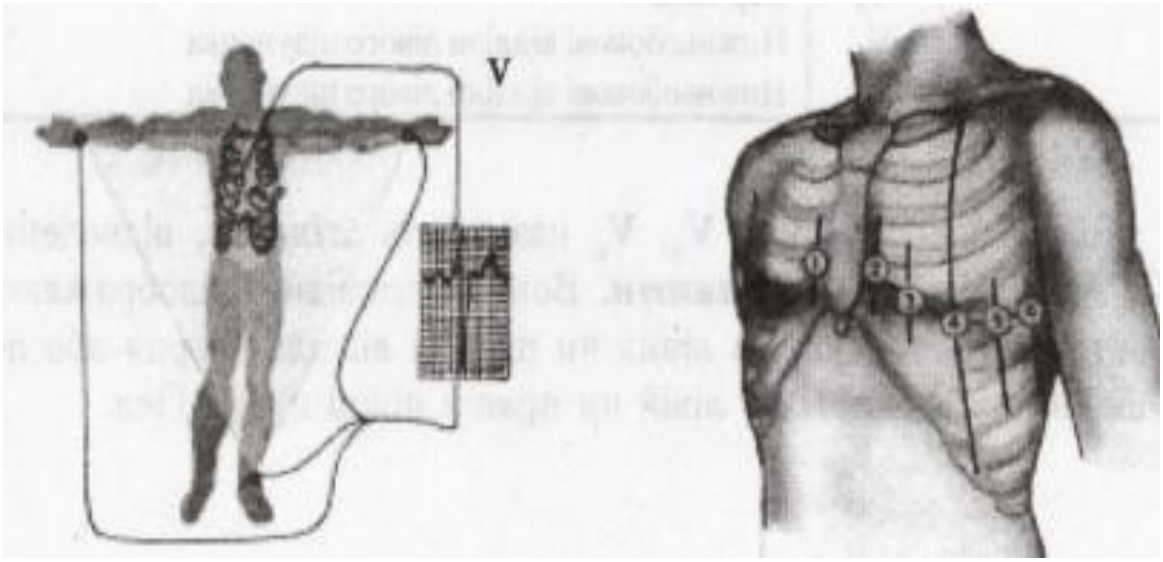

Рисунок 1.3 – Грудні відведення Вільсона

Грудні відведення Вільсона включаються в себе наступні відведення:

- V — активний електрод (білий) – IV м/р справа біля груднини;
- V2 — активний електрод (білий) – IV м/р зліва біля груднини;
- V3 — активний електрод (білий) – посередині прямої між v2-v4;
- V4 — активний електрод (білий) –.ділянка верхівки,VM/P по серединно-ключичній лінії;
- V5 — активний електрод (білий) – V м/р по передній аксилярній лінії зліва;
- V6 — активний електрод (білий) – V м/р по середній аксилярній лінії зліва [6].

Електрокардіограма (ЕКГ) візуально представлена у вигляді кривої із 5 зубцями. Всі вони позначаються латинськими літерами – P, Q, R, S, T (рисунок 1.4), та кожен зубець відображає певну стадію збудження міокарда:

- зубець Р виникає при порушенні передсердь,
- QRS-комплекс виражає скорочення шлуночків,
- зубець Т являє собою вихід міокарду зі стану збудження.

Вид ЕКГ може відрізнятись в залежності від відведення, на якому фіксується значення напруги. Про патологічні зміни можуть свідчити будь-які відхилення на кардіограмі — зміна ширини та/або висоти зубців, їх присутність/відсутність, співвідношення, періодичність і т.д. Навіть найменш помітні аномалії можуть бути зафіксовані лікарями при обстеженні та враховані при визначенні діагнозу стану здоров’я хворого, визначенні діагнозу.

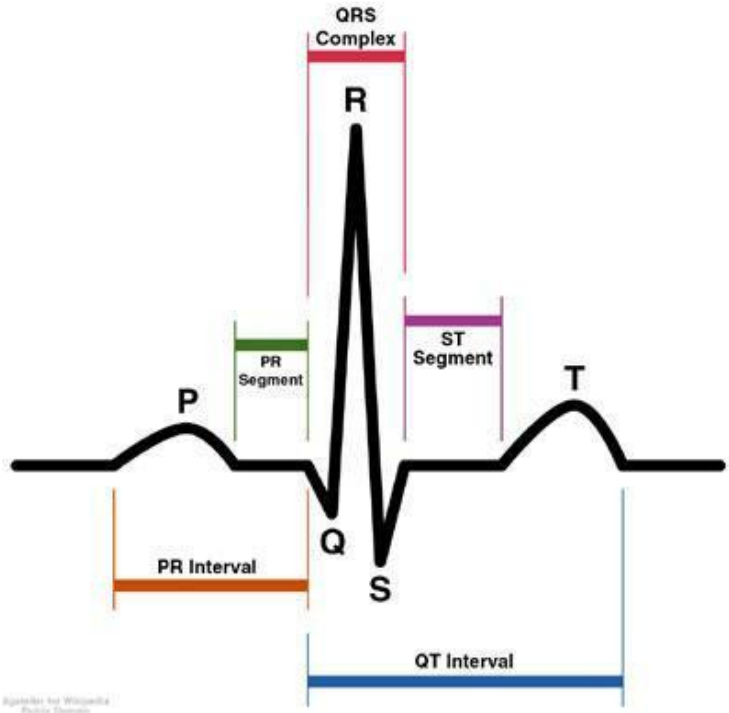

Рисунок 1.4 – Основний паттерн кардіограми (один удар серця) — зубець P, QRS-комплекс, зубець T

З точки зору аналізу кардіограм, з ЕКГ можна отримати такі види сигналів:

Сирий сигнал – найбільш інформативний, не стиснутий сигнал, що було знято з пацієнта без подальшої обробки.

RR-інтервали – проміжки часу між ударами серця (між сусідніми R-піками електрокардіограми). Одним із способів визначення ряду RR-

інтервалів є просвічування шкіри (фотоплетизмографія) [8]. Дуже часто аналіз ритмограм (ряду RR-інтервалів) використовують для визначення впливу навантаженості нервової системи на серце [9].

Анотовані дані – це координати ключових точок кардіограми -- частин QRS-комплексу, координати початку/кінця P- та T-хвиль, тощо. Приклад таких даних показано на рисунку 1.4.

Якість отриманих даних може залежати від впливу багатьох факторів, таких як поза людини, її втомленість/застресованість, впливу хімічних речовин, точності апаратури, тощо [11].

1.2 Сучасні способи зняття кардіограми

На даний момент електрокардіограми зазвичай записують, спираючись на принципи, що описані вище. Сучасні медичні кардіографи можна поділити на наступні групи (рисунок 1.5):

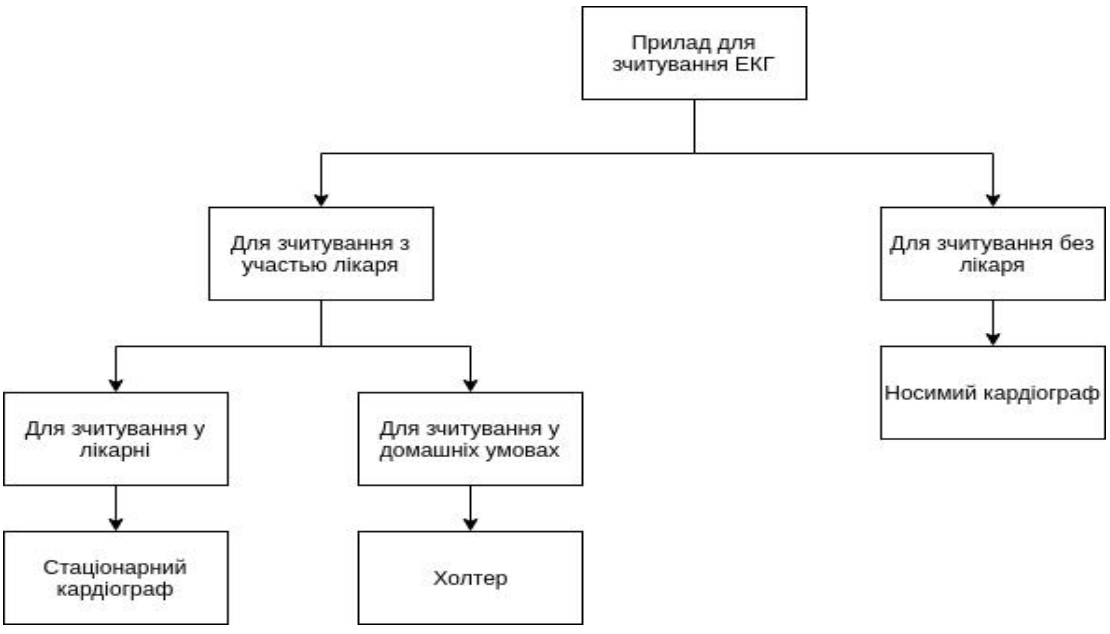

Рисунок 1.5 – Класифікація сучасних приладів для зчитування ЕКГ

Стаціонарний кардіограф найчастіше використовуєть в лікарнях, при відстеженні ЕКГ у пацієнтів у тяжкому стані, для максимально докладного аналізу. Ці кардіографи використовують інформацію, отриману з усіх 12 відведень. У зв'язку із складністю та розмірами системи, найбільш точний аналіз кардіограм у реальному часі можливий лише за стаціонарних умов.

Холтер – медичний портативний кардіограф. Він має вдвічі менше відведень, ніж стаціонарний, тому забезпечує менш високу точність. Використовується в лікарнях для проведення моніторингу стану пацієнта протягом певного проміжку часу та часто в русі (наприклад, при виконанні вправ). Суттєвою перевагою таких приладів є можливість переносу записів ЕКГ на жорсткий диск для збереження та подальшого розповсюдження.

Носимий кардіограф – найменші за розмірами кардіографи з найменшої кількістю відведень (зазвичай усього 1-3 відведення). Їх розміри і швидкість передачі даних – їх найбільша перевага, зазвичай використовуються для автоматизованого аналізу ЕКГ, знятою протягом великих проміжків часу. Працюють за декількома принципами:

- моніторинг ЕКГ для відправки лікарю;
- моніторинг ЕКГ для автоматизованого аналізу.

1.3 Види ЕКГ

Для того, щоб вирішувати задачу класифікації сигналів ми маємо розуміти можливу природу шумів. Тоді при створенні необхідної комбінації методів та алгоритмів буде можливість враховувати максимальну кількість як позитивних, так і негативних паттернів. Розглянемо один із варіантів класифікації сигналів ЕКГ за зашумленністю (рисунок 1.6).

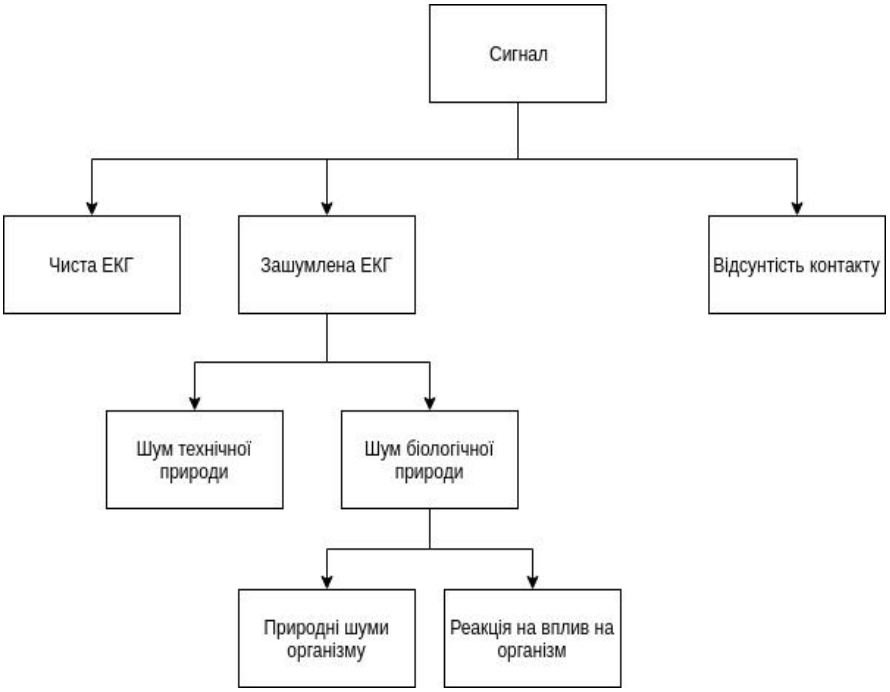

Рисунок 1.6 – Класифікація ЕКГ сигналу за зашумленістю

1.4 Відкриті датасети медичних кардіограм

Для якісної роботи алгоритмів з використанням методів машинного навчання та нейронних мереж, необхідно мати якомога більший датасет, оскільки із збільшенням навчальної вибірки підвищується точність розпізнавання та досліджується більший розмах можливих аномалій.

Розглянемо кілька прикладів найвідоміших баз кардіограм

The ECG-ID Database – база кардіограм, що містить 310 записів отриманих з 90 людей. Кожний запис містить:

- кардіограму з 1-го відведення, довжиною 30 секунд та із частотою дискретизації 512 Гц;
- 10 аннотованих точок;
- інформація у .hea файлі, що містить вік, стать, дату запису.

Записи кардіограми були отримані від добровольців (44 чоловіки та 46 жінок віком від 13 до 75 років). Для кожної людини у базі присутні від 2 до 20 записів, що були зібрані періодично. [26].

Сирі сигнали ЕКГ є не лише найбільш інформативними, але шумними – в них можуть міститись компоненти високо-, так і низькочастотних шумів. Кожен запис включає як сирі, так і відфільтровані сигнали:

- signal 0: ECG I (сирий сигнал);
- signal 1: ECG I фільтрований (фільтрований сигнал).

MIT-BIH Arrhythmia Database – зібрана у 1975 році бостонською лабораторією. Ця база даних вважається класичною, оскільки у свій час стала першою загально доступною БД із матеріалами для базових досліджень динаміки серця.

MIT-BIH Arrhythmia DB містить 48 півхвилинних витягів з двоканальних амбулаторних записів ЕКГ, отриманої з 47 піддослідних, вивчених лабораторією Arrhythmia BIH в період з 1975 по 1979 рр. Двадцять три записи були обрані випадковим чином з набору з 4000 24- годину амбулаторних записів ЕКГ, зібраних із змішаної кількості населення стаціонарних (близько 60%) та амбулаторних (близько 40%) у Бостонському лікарні Бет-Ізраїль; решта 25 записів були відібрані з того самого набору, що включає менш поширені, але клінічно значущі аритмії, які не були б добре представлені у невеликій довільній вибірці [16].

Записи були оцифровані на 360 зразках в секунду на канал з роздільною здатністю 11 біт на 10 мВ діапазоні. Два або більше кардіологів самостійно коментують кожен запис; Розбіжності були вирішені, щоб отримати автоматично читаємі довідкові анотації для кожного удару (приблизно 110 000 анотацій у всіх), включених до бази даних [27].

РОЗДІЛ 2 МЕТОДИКА ВИКОРИСТАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ ВИЗНАЧЕННЯ ФОРМ ЕКГ

У цьому розділі будуть розглянуті різні методи, що можуть бути використані для класифікації та кластеризації електрокардіограм, засновані на застосуванні мереж навчання з учителем, в тому числі нейронних мереж і методу опорних векторів. Будуть проілюстровані різноманітні методи попередньої обробки даних: вейвлет-аналіз, статистики вищого порядку (СВП), Ермітівська характеристика QRS-комплексу реєстрованої хвилі ЕКГ.

Для досягнення найвищого точності класифікації пропонується комбінувати різні методи класифікації та вибору ознак, за принципом так званого «зваженого голосування». Цей принцип буде розглянуто на прикладі класифікатора, що спирається на метод опорних векторів – ваги регулюються відповідно до результатів роботи окремих класифікаторів на навчальній вибірці даних. Робота проводитиметься із даними із MIT-BIH Arrhythmia Database [10].

2.1 Виділення характеристик

Розпізнавання і класифікація образів, включаючи сигнали кардіограм, вимагають виділення характеристик [7], що точно описують ці образи з метою диференціювати їх типи або класи.

Ці характеристики представляють образи так, що відмінності в морфології хвиль ЕКГ придушуються для однакових типів (класів) серцевих скорочень і посилюються для форм хвиль, що належать до різних типів скорочень. Це перевірено на формах хвиль ЕКГ з бази MIT-BIH Arrhythmia

Database [10]. У цій базі міститься 12 типів аномалій та порушень серцевих скорочень: блокада лівої ніжки пучка Гіса (L) , блокада правої ніжки пучка Гіса (R), передсердна екстрасистола (A), аберрантним проведена передсердна екстрасистола (a), вузлова екстрасистола (J), шлуночкова екстрасистола (V), злиття шлуночкового і нормального скорочення (F), мерехтіння шлуночків з ердца (I), вузлове запізнюється скорочення (j), запізнюється скорочення шлуночків (E), надшлуночкова екстрасистола (S), «зливний» комплекс (злиття викликаного і спонтанного скорочень при штучної електрокардіостимуляції) (f), комплекси, що відповідають нормальному синусовому ритму (N) [8]. Зразкові форми хвиль ЕКГ одного пацієнта [10], відповідні нормального ритму (N) і трьох типів пошкоджень ритмів (L, R і V). Значення по вертикальній вісі у вимірюється в мВ, а по горизонтальній – при частоті дискретизації 360 Гц одна точка відповідає приблизно 2,8 мс.

Відомо, що існує великий розмах в морфології серцевих скорочень, що належать до одного класу, навіть для одного пацієнта. Більш того, скорочення, що належать до різних класів, схожі за структурою один на одного (це можна помітити, порівнявши ритми L-типу і V-типу).

Вони знаходяться у практично однакових областях частот і амплітуд, тому може бути тяжко відрізнити один ритм від іншого, спираючись лише на часове та/або частотне подання. В такому випадку хорошою практикою є використання характеристик, які описують морфологію QRS-комплексу, такі як ширина і параметри хвилі, параметри RR-інтервалів і т.д. [6]. Досить часто характеристики отримують, використовуючи перетворення Фур'є [2] або вейвлет-перетворення [11]. Також іноді може бути корисна кластеризація даних електрокардіограм за допомогою СОК [3] або КВО [1], внутрішні характеристики, отримані на етапі попередньої нейронної обробки сигналів [1].

Інші методи виділення характеристик працюють із статистичними показниками [5]. Жоден з цих методів, звичайно, не буде ідеальним. Далі буде

проілюстровано метод класифікації з учителем, заснований на обробці характеристик, отриманих з опису QRS-комплексу, за допомогою функції розкладання Ерміта.

2.2 Функція розширення Ерміта

У методі Ерміта QRS-комплекс представлz'nmcz у вигляді набору функцій Ерміта. Цей підхід успішно використовує існуючу схожість між формами функцій розкладання Ерміта та QRS-комплексами хвилі ЕКГ, що аналізується. Також ця характеристика використовує параметр ширини комплексу, який дає багато інформації про скорочення з великою різницею в тривалості QRS-комплексу. Позначимо QRS-комплекс хвилі ЕКГ через *x(t)*. У серії функцій Ерміта це можна записати як:

$$x(t) = \sum_{n=0}^{N-1} c_n \varphi_n(t, \sigma),$$

де c_n це коефіцієнт розширення, σ це параметр ширини, а $\phi_n(t, \sigma)$ це функція розкладання Ерміта n -го порядку, що визначається як [3]:

$$\varphi_n(t, \sigma) = \frac{1}{\sqrt{\sigma 2^n n! \sqrt{\pi}}} e^{-t^2/2\sigma^2} H_n(t/\sigma)$$

де $H_n(t/\sigma)$ це поліном Ерміта n –го порядку. Поліноми Ерміта задовольняють наступному рекурентному співвідношенню:

$$H_n(x)=2xH_{n-1}(x)-2\,(n-1)H_{n-2}(x)$$

при цьому $H_0(x) = 1$ і $H_0(x) = 2x$ для $n = 2,3, \ldots$. Чим вище порядок полінома Ерміта, тим вище частота його змін в часовій області і тим вище його здатність характеризувати швидкі зміни в сигналі ЕКГ. Коефіцієнти c_n функції розкладання Ерміта можуть бути отримані шляхом мінімізації адитивної квадратичної помилки:

$$E = \sum_{I}\left[x(t_i) - \sum_{n=0}^{N-1} c_n\varphi_n(t,\sigma)\right]^2$$

Ця функція помилки являється набом лінійних рівнянь по відношенню до коефіцієнтів c_n. Ці рівняння можна розв'язати за допомогою сингулярного розкладання і методу псевдообернення [12]. У численних обчисленнях сегмент QRS сигналу ЕКГ представлений 91 точкою даних навколо R піку (45 до і 45 після). Частота дискретизації 360 Гц утворює вікно тривалістю 250 мс, якого достатньо для типового QRS-комплексу. Дані додатково розширюються шляхом додавання 45 нулів до кожного сегменту QRS.

Віднімання середнього значення першої і останньої точок нормалізує сигнал ЕКГ. Параметр ширини σ був обраний пропорційно ширині QRS-комплексу. Ці модифіковані QRS-комплекси ЕКГ були перетворені в лінійну комбінацію функцій Ерміта. Більш глибокий аналіз показав, що 15 коефіцієнтів Ерміта дозволяють досить добре відтворити криві QRS з метою зобразити найважливіші деталі цих кривих [3]. На рисунку 2.1 зображено приклад нормалізованого за допомогою 15 функцій Ерміта QRS-комплексу. Горизонтальна вісь графіку вимірюється в точках.

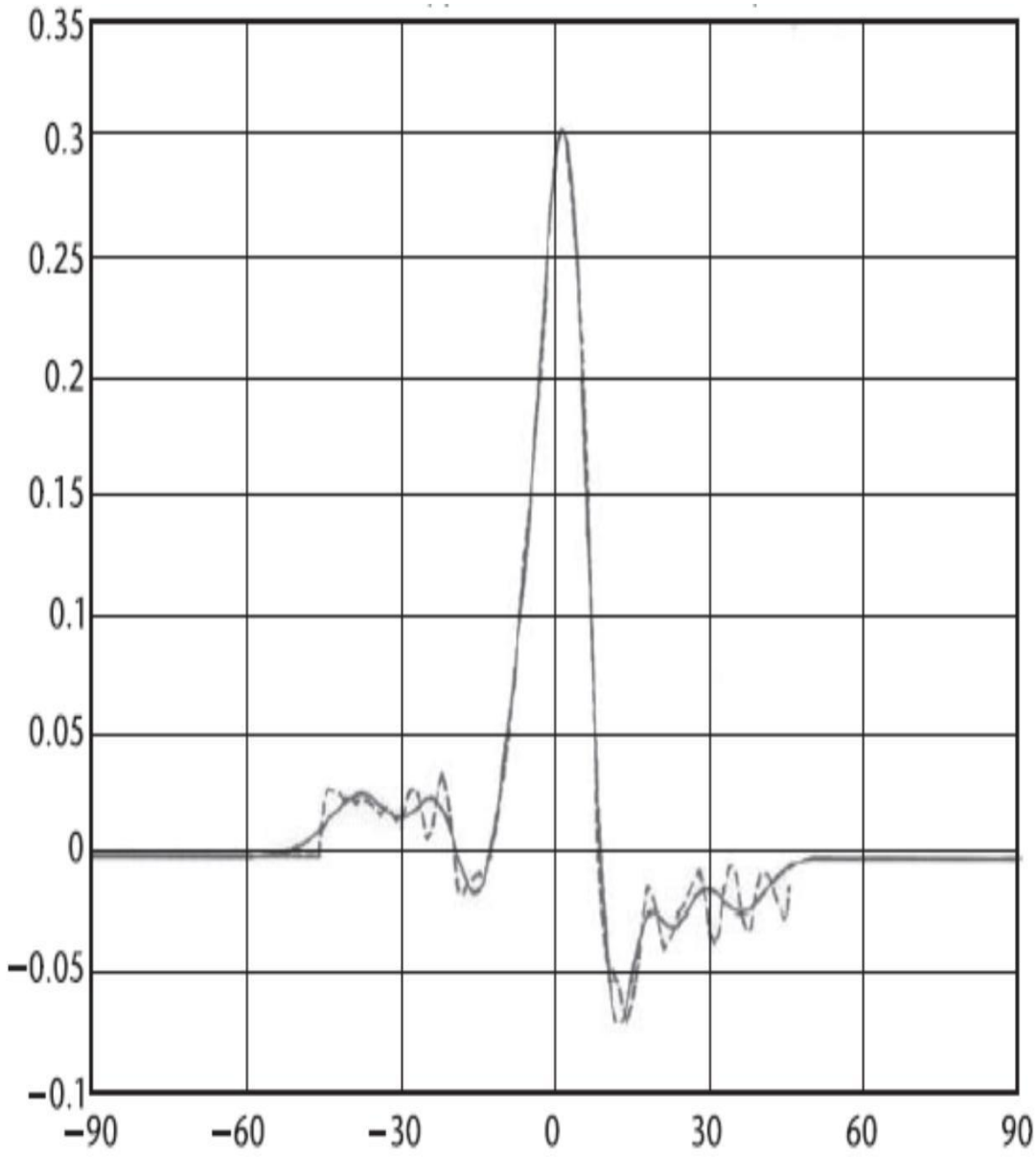

Рисунок 2.1 - Апроксимація QRS-комплексу за допомогою 15 функцій Ерміта

Ці коефіцієнти, разом з 2 класичними характеристиками сигналу (довжиною RR-інтервалу і середнім RR-інтервалом попередніх 10 скорочень) формують вектор x , що складається з 17 елементів і подається на вхід класифікатора. Ці дві характеристики зазвичай вибираються для кращого уявлення фактично оброблюваного сегмента хвилі на тлі середньої довжини останніх оброблених сегментів.

2.3 Статистики більш високого порядку як характеристики ЕКГ

Іншим важливим підходом для генерації характеристик ЕКГ є застосування статистичних дескрипторів хвиль QRS. Застосовуються 3 типи статистик: полуінваріанти 2-го, 3-го і 4-го порядків. Полуінваріанти - це коефіцієнти розкладання в ряд Тйлора навколо s=0 для функції генерації полуінваріантов змінної x, розраховуються як $\varphi_x(s)=ln\{E[esx]\}$, де E це оператор математичного сподівання [13].

Вони можуть також бути виражені з точки зору добре відомих статистичних моментів, а саме як їх комбінація – лінійна або нелінійна. Для стаціонарного процесу $x(t)$ із середнім значенням рівним нулю, полуінваріанти 2-го і 3-го порядку дорівнюють відповідним їм моментам:

$$c_{2x}(\tau_1)=m_{2x}(\tau_1)$$
$$c_{3x}(\tau_1,\tau_2)=m_{3x}(\tau_1,\tau_2)$$

Момент n-го порядку для $x(k)$, $m_{nx}(\tau_1, \tau_2,...,T_{n-1})$, формально визначається [13] як коефіцієнт навколо s= 0 в розкладанні Тейлора для функції генерації моменту $\varphi_x(s)$, де $\varphi_x(s) = E[e^{sx}]$. Аналогічно, кожен статистичний момент n-го порядку може бути розрахований шляхом взяття математичного сподівання процесу помноженого на самого себе з затримкою (n-1).

Вираз для полуінваріанта 4-го порядку вираховується трохи складніше [13]:

$$c_{4x}(\tau_1,\tau_2,\tau_3) = m_{4x}(\tau_1,\tau++_2,\tau_3)-m_{2x}(\tau_1)m_{2x}(\tau_3-\tau_2)$$
$$-m_{2x}(\tau_2)m_{2x}(\tau_3-\tau_1)-m_{2x}(\tau_3)m_{2x}(\tau_2-\tau_1)$$

У цьому виразі c_{nx} це полуінваріант n-го порядку, а m_{nx} це статистичний момент n-го процесу $x(k)$. τ_1, τ_2 і τ_3 це тимчасові затримки.

Були обрані значення інваріантів 2-го, 3-го і 4-го порядків в п'яти точках, рівномірно розподілених по довжині QRS-комплексу (для інваріантів 3-го і 4-го порядку були використані діагональні шматки) як характеристики, що використовуються для практичного прикладу розпізнавання серцевого ритму. П'ять точок було вибрано достижени я схеми кодування ознак порівнянної з поданням Ерміта.

Для вектора, що описує QRS-комплекс і складається з 91 елемента, були обрані полуінваріанти, відповідні тимчасових затримок в 15, 30, 45, 60 і 75. Додатково були обрані дві часові характеристики: одна відповідає поточному RR-інтервалу скорочення, а друга являє собою середній RR-інтервал, тривалість якого дорівнює 10 попереднім скорочень.

У такому випадку кожне скорочення має вигляд вектора характеристик з 17 елементів, перші 15 з яких відповідають СВП QRS-комплексу (полуінваріанти 2-го, 3-го і 4-го порядку, що складаються з 5 елементів кожен), а останні 2 це часові характеристики поточного QRS-сигналу. Застосування полуінваріантних характеристик QRS-комплексу знижує відносний розподіл характеристик ЕКГ, що належать до одного типу серцевого ритму і робить класифікацію щодо більш простий. Це добре видно на прикладі дисперсії сигналів, що відносяться до нормальних (N) і анормальним (L, R, A, V, I, E) скорочень.

У таблиці 1 представлена дисперсія для обраних 7 типів нормалізованих серцевих скорочень і їх полуінваріантние характеристики для більш ніж 6600 скорочень з бази MIT-BIH AD [10].

Це доводить, що дисперсія полуінваріантних характеристик значно знижується по відношенню до дисперсії вихідних сигналів.

Таблиця 2.1 - Дисперсія обраних серцевих ритмів з MIT-BIH AD

Тип ритму	Початковий сигнал QRS	Полуінваріант 2-го порядку	Полуінваріант 3-го порядку	Полуінваріант 4-го порядку
N	0,74E-2	0,31E-2	0,28E-2	0,24E-2
L	1,46E-2	0,60E-2	1,03E-2	0,51E-2
R	1,49E-2	0,94E-2	1,06E-2	0,55E-2
A	1,47E-2	0,67E-2	0,85E-2	0,38E-2
V	1,64E-2	0,68E-2	0,71E-2	0,54E-2

Це означає, що розподіл значень параметрів, що характеризують сигнали ЕКГ, що належать до одного класу, тепер менше, що робить процес розпізнавання набагато простішим. Цей феномен був підтверджений великою кількістю обчислювальних експериментів для всіх типів серцевих скорочень з бази MIT-BIH AD.

2.4. Вейвлет перетворення

Для згладжування ЕКГ та фільтрації сигналу використовуємо вейвлет-перетворення (рис 2.2) . Вони схожі на перетворення Фур’є, але значно краще працюють на нестаціонарних рядах.

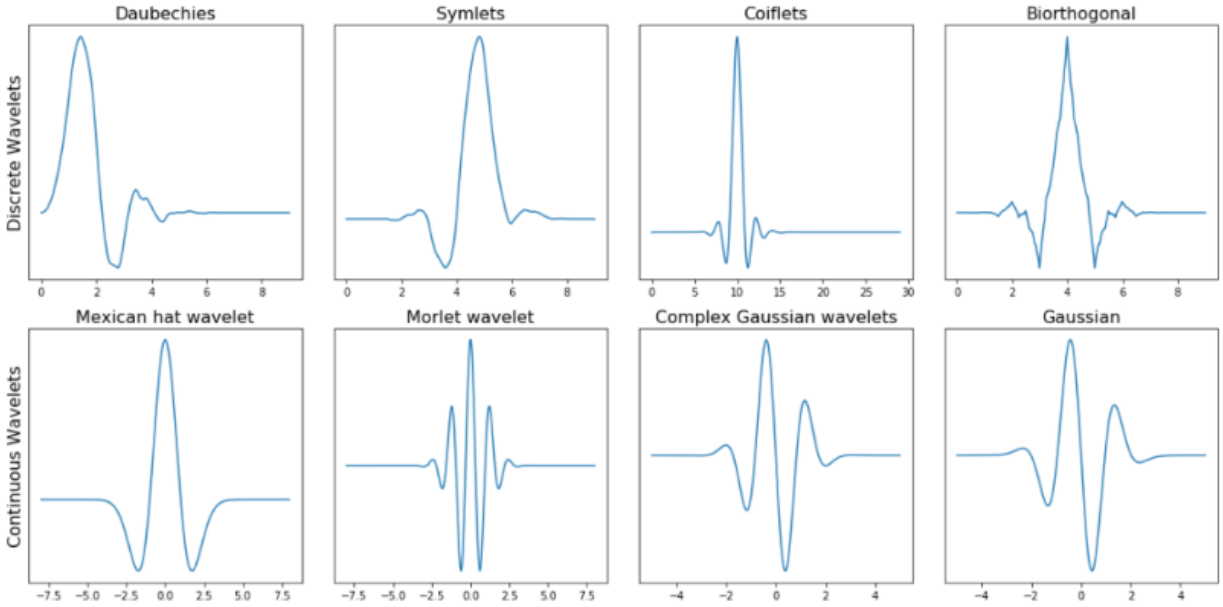

Рис. 2.2 – Приклади неперервних та дискретних вейвлетів

Алгоритм виділення коефіцієнтів розкладу наведено на рис. 2.3 - 2.5

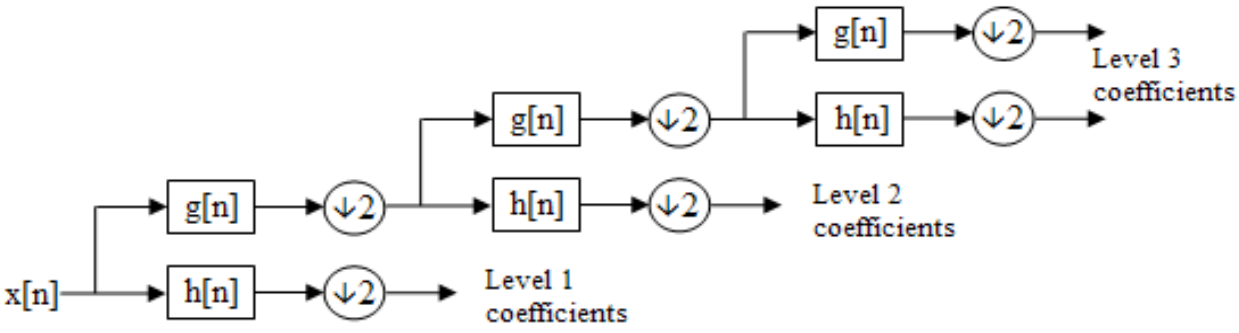

Рис. 2.3– Розклад по рівням, приклад алгоритму

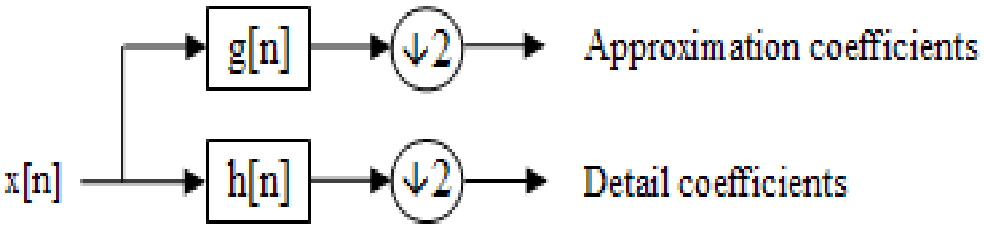

Рис. 2.4 – Коефіцієнти розкладу

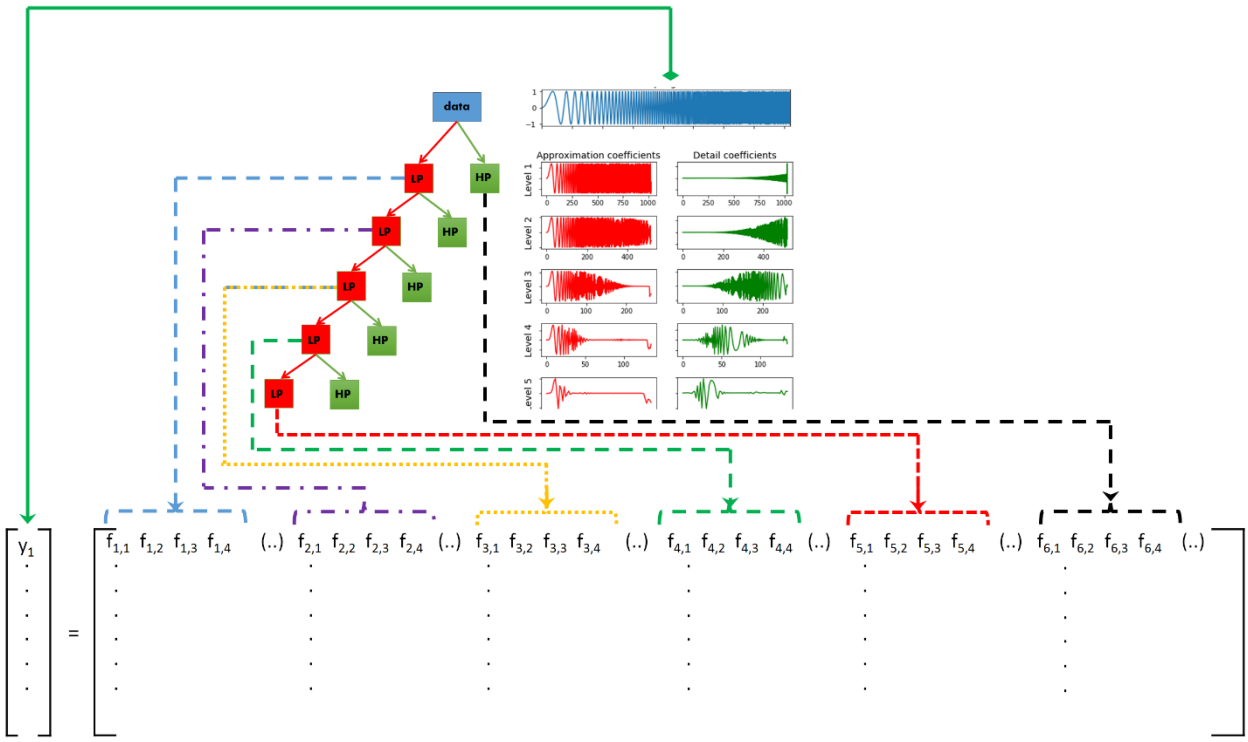

Рис. 2.5 – повна схема розкладу.

2.5. Нейронна класифікація з навчанням

Компоненти вхідного вектора x, що містять характеристики ЕКГ, описують вхід класифікатора. Нейронні класифікатори з навчанням вважаються одними з найбільш ефективних класифікаторів [7, 14, 15]. Зупинимося на наступних моделях: МПЦ, гібридні нечіткі мережі, нейро-нечіткі мережі Такагі-Сугено-Канга, а також метод опорних векторів, котрий є підтипом нейронної системи [14]. В основі процесу навчання всіх цих моделей лежить набір навчальних пар (x_i, d_i), де x_i це вектор характеристик, а d_i це вектор кодів класу.

2.5.1 Багатошаровий перцептрон

Багатошаровий перцептрон [14, 16] це одна з найбільш відомих нейронних мереж, що складається з безлічі простих нейроноподібних модулів обробки сигмовидної функції активації, згруповані в шари. Функції активації в багатошаровому перцептроном можуть мати різну форму: логарифмічну $f(x)=1/(1+\exp(-x))$, гіперболічного тангенсу, сігнуму (кусково-постійної функції) або лінійну. Класична мережа складається з одного прихованого шару, за яким слідує вихідний шар нейронів.

Інформація локально обробляється в кожному модулі шляхом обчислення скалярного твори відповідного вхідного вектора і вагового вектора нейрона. Перед тренуванням вагові вектора створюються випадковим чином. Тренування мережі для отримання бажаного вихідного вектора d_i, коли відомий вхідний вектор x_i, звичайно включає систематичне зміна вагових векторів всіх нейронів до тих пір, поки мережа не видасть бажаний результат на виході, забезпечуючи при цьому допустиму похибку (помилку). Це повторюється протягом усього циклу тренування. Таким чином, навчання зводить до мінімуму процес вимірювання помилки для всього навчального набору даних протягом кінцевого числа циклів навчання для запобігання перенавчання [14].

Найбільш ефективні методи навчання засновані на градієнтних моделях. Градієнтні вектора в багатошаровій мережі розраховуються з використанням алгоритму зворотного поширення помилки [14]. У градієнтному методі навчання вагові вектори w підбираються від циклу до циклу відповідно до інформації градієнту функції помилки

$$w(k+1)=w(k)+\eta p(k),$$

де η - це константа навчання, що розраховується в кожному циклі, а $p(k)$ - це спрямовуючий вектор мінімізації в k-му циклі. Після закінчення етапу тренування знайдені вагові вектора «заморожуються» і потім використовуються в тестовому режимі, в якому вхідний вектор x , оброблений мережею для освіти вихідного нейронного сигналу, відповідає за визначення класу. Зазвичай нейрон найбільшого вихідного сигналу відповідає обумовленому класу.

Узагальнення це фундаментальна властивість, що дозволяє визначити здатність нейронної мережі до розпізнавання образів, що не входять в тренувальний набір. Якщо кількість ваг мережі занадто висока або кількість тренувальних прикладів дуже мала, то буде величезна кількість мереж, відповідних тренувальних даними, але лише деякі будуть точно описувати простір вірних рішень. Отже, більш переважно слабке узагальнення. Щоб підвищити ймовірність правильного узагальнення, необхідно мінімізувати кількість вільних параметрів (ваг). Однак це потрібно зробити так, щоб розмір мережі не зменшився до області, в якій вже не досягти бажаного результату. Більш того, необхідно контролювати кількість циклів навчання, щоб уникнути надмірного відповідності моделі тренувальним даними (перенавчання мережі).

Інший спосіб - перехресна перевірка на достовірність (перехресна валідація). У цьому методі дані розбиваються на тренувальний, зразковий і тестовий набори. Набір зразкових даних використовується для перевірки здатності мережі, навченої на тренувальних даних, до узагальнення. Розмір мережі, що забезпечує мінімальну помилку валідації, приймається за оптимальнийий.

2.5.2 Рекуррентні мережі

Ідея RNN (рис. 2.6.) полягає у послідовному використанні інформації, навідміну від традиційних нейронних мереж. Для багатьох задач підхід, в якому вважається, що всі входи та виходи незалежні, не підходить – якщо ви хочете визначити поведінку часового ряду або передбачити наступне слово в тексті, необхідно брати до уваги попередні елементи послідовності. Власне, саме тому рекуррентні мережі і звуться рекуррентними – оскільки вони виконують одну і ту саму задачу для кожного елементу послідовності, причому в залежності від значень, отриманих на попередніх кроках (кількість кроків визначається в залежності від задачі).

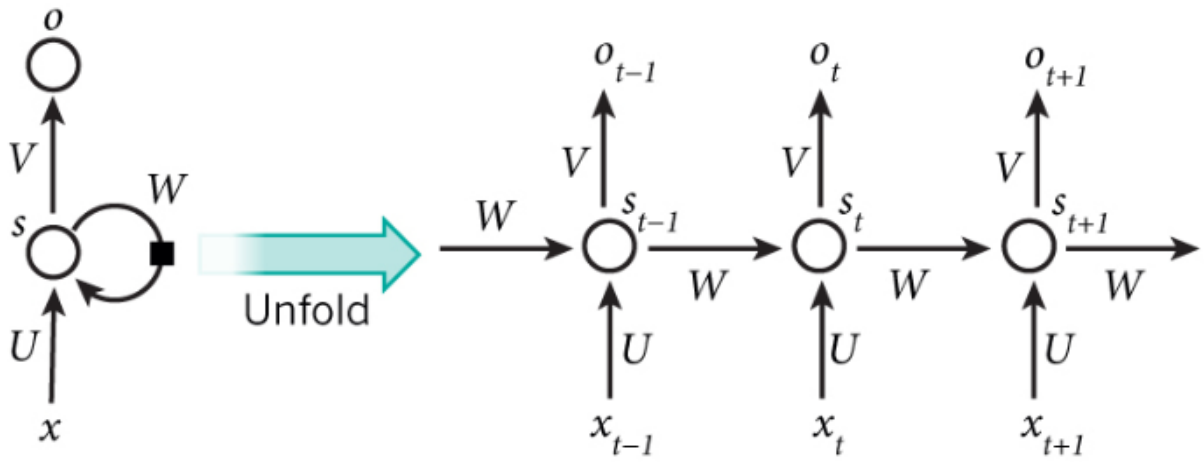

Рис. 2.6 – Рекуррентна мережа та її розгортка

Навчання RNN аналогічне навчанню звичайної нейронної мережі – так само використовується алгоритм backpropagation, але із невеликою зміною – градієнт на кожному виході залежить не лише від поточних розрахунків, але й від попередніх кроків, тому необхідно «протягнути» помилку на три кроки назад і просумувати градієнти. У цього підходу є суттєвий недолік – в ньому виникають суттєві складності з вивченням довгострокових залежностей, де

розрив між інформацією та місцем її потенціальної потреби завеликий. Для запобігання цьому було розроблено кілька спеціальних архітектур, таких як LSTM та BNN.

BNN (Bidirectional RNNs) – двонаправлені нейронні мережі. Ідея їх базується на тому, що вихід в деякий момент часу може залежати не лише від попередніх значень, але й від майбутніх. Реалізація подібна до двох RNN, покладених одна на одну (рис. 2. 7) , а вихід розраховується на основі стану обох. Цю модель можна ще ускладнити додаванням більшої кількості рівней на кожний крок часу, але це потребує ресурсів.

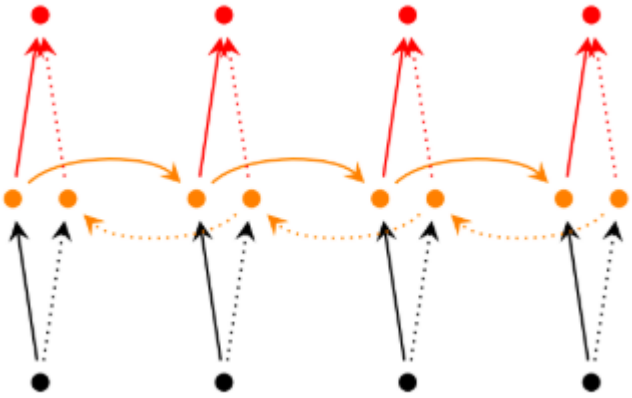

Рисунок 2.7 – структура BNN

LSTM – тип рекурентної мережі, що може вивчати довгострокові залежності. Вони розроблені з метою запам’ятовування інформації протягом довгих періодів часу. Ці мережі не суттєво відрізняються структурою (рис. 2.?, рис. 2.?) , але вони використовують іншу функцію для обчислень станів.

Всі мережі мають форму «ланцюжка» повторюваних модулей. У стандартних RNN цей повторюваний модуль має просту структуру – наприклад, один шар tanh (рис. 2.8). LSTM має всередині три «гейти» для контролю стану комірки – шар втрати, шар збереження і шар оновлення (рис. 2.9).

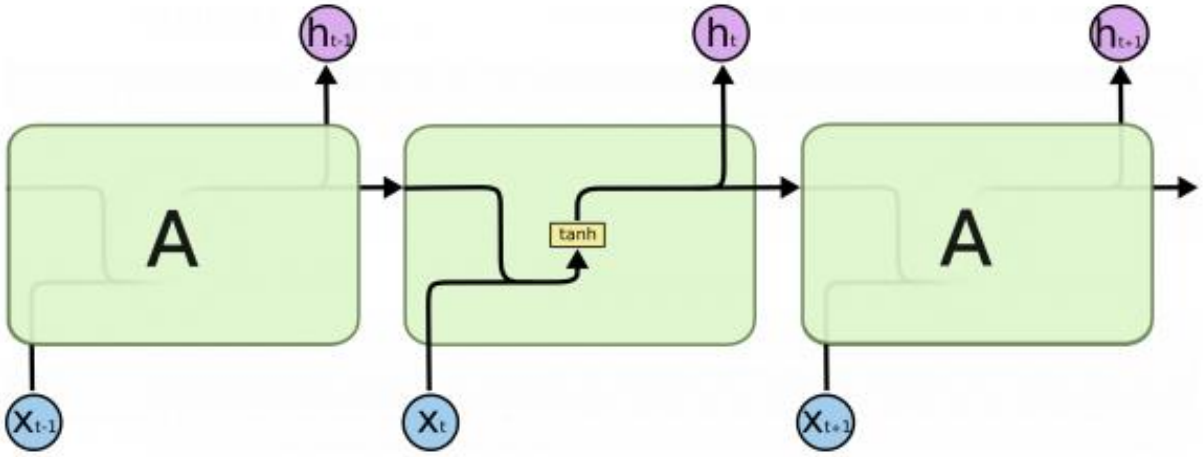

Рис 2.8 – структура модулю RNN

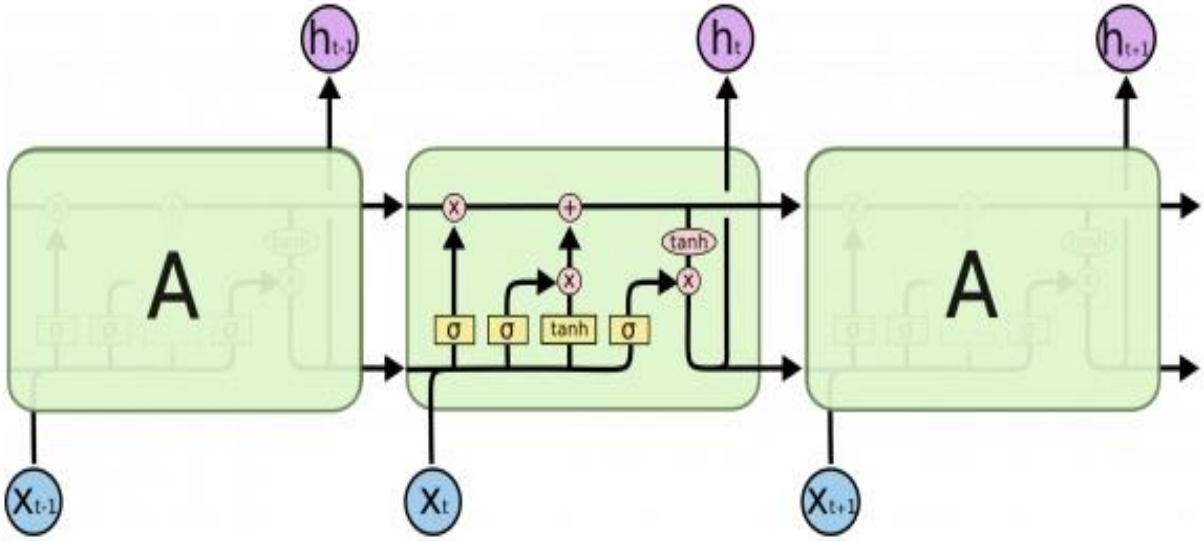

Рис. 2.9 – структура модулю LSTM

2.5.3 Згорткові мережі

Згорткові мережі (рис. 2.10) працюють на основі фільтрів, що займаються розпізнаванням деяких конкретних характеристик зображення. Фільтр є колекцією вагів, які навчаються з метою навчитись визначати ці

характеристики. Фільтр переміщується протягом зображення і визначає, чи присутня деяка характеристика в поточній частині зображення. А реалізується це за допомогою операції згортки – суми добутків елементів фільтра та матриці вхідних сигналів.

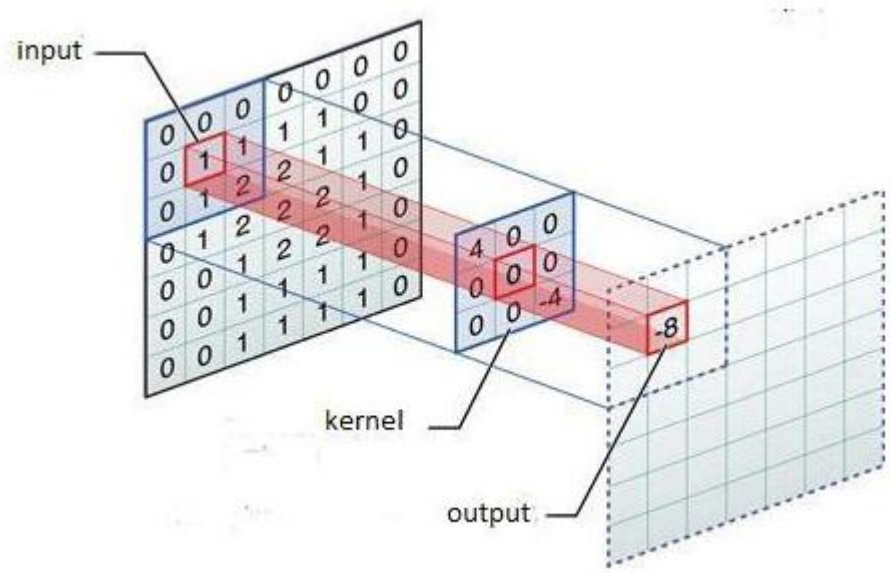

Рис. 2.10 – Операція згортки

Якщо характеристика присутня – фільтр видасть велике число, якщо відсутня – відносно мале (часто 0). Результатом обробки фільтром є матриця, яка складається із результатів одиничних згортов, які були фільтром породжені.

Загалом, ЗНМ складаються із вхидного, виідного та приховаинх шарів. Приховані шари мають в собі шари згортки, агрегації, з’єднання і нормалізації.ЗНМ часто потребують меншої обробки даних, ніж альтернативні методи – в цьому велика переваги цих мереж. Що цікаво, на їх створення дослідників надихнула сама природа, оскільки схема зв’язків між нейронами нагадує схему організації зорової системи у тварин.

2.5.4 Гібридні нечіткі мережі

Гібридні нечіткі мережі - це комбінація каскадно-пов'язаних нечітких систем, самоорганізованих шарів і БП як показано на рисунку 1.8.

Це узагальнення так званої мережі зустрічного поширення Хет Нільсена. Замість використання шарів Конохена ця модель застосовує нечіткі самоорганізованих шари, а підмережа БП (з одним прихованим і одним вихідним шаром) використовується замість шару Гроссберга [13].

Нечіткі самоорганізовані шари відповідають за нечітку кластеризацію вхідних данн их, в яких вектор x спочатку класифікований в усі кластери з різними класами елементів. Кожен вхідний вектор x_j в майбутньому ставиться до різних кластерам з центром c_i і значенням елемента $\mu_i(x_j)$, що визначається як:

$$\mu_i(x_j) = \frac{1}{\sum_{k=1}^{K}(d_{ij}/d_{kj})^{\frac{2}{m-1}}},$$

де K це кількість кластерів, а d_{kj} - відстань між j-м вхідним вектором x_j і k-м центром c_k. Кількість кластерів зазвичай є більшою за кількість класів.

Застосування нечіткої кластеризації дозволяє домогтися кращої фільтраційної здатності простору даних. У цій моделі положення вхідного вектора x в багатовимірному просторі визначено більш точно. Це необхідно для реалізації системи розпізнавання серцевих скорочень, де вектора асоціюються з різними класами, що використовують схожий спектр параметрів.

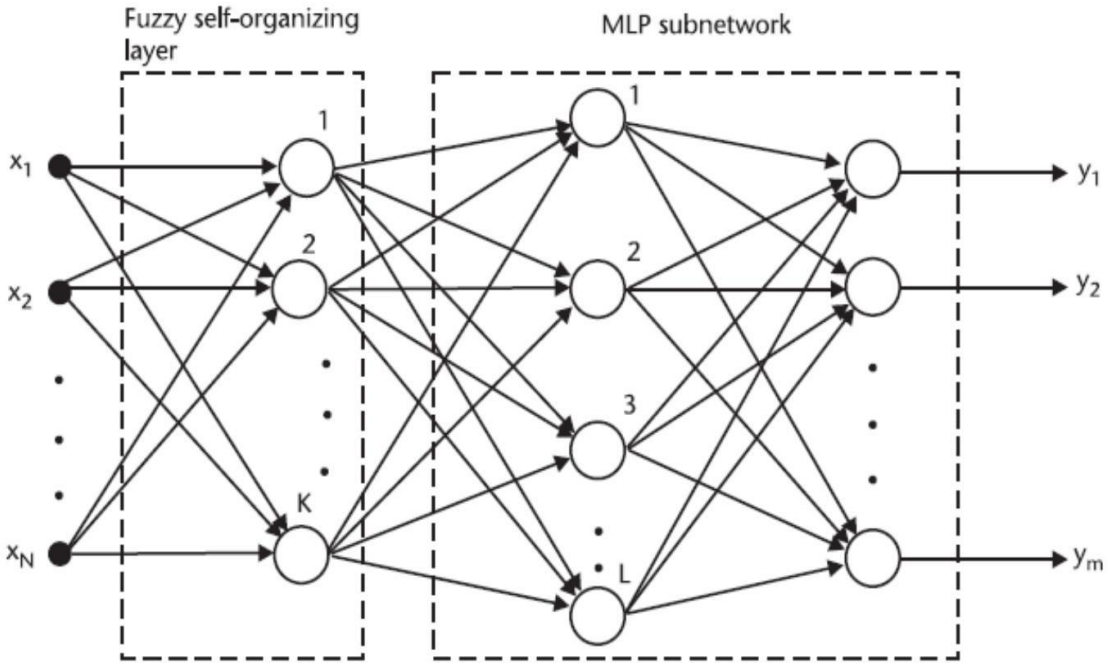

Рисунок 1.8 - Узагальнена структура нечіткої гібридної мережі

Сигнали нейронів, що самоорганізуються, представляють кластер класів елементів $\mu_i(x_j)$, формують вхідний вектор для другої підмережі МП. МП відповідає за звичайно е відповідність між вхідними сигналами і допустимим класом (кінцева класифікація). Ця підмережа навчається після того як отримано перший самоорганізується шар. Алгоритм навчання ідентичний тому, що використовується для навчання МП.

2.5.5 Метод опорних векторів

МОВ - це вид мережі з позитивним зворотним зв'язком, запропонованої Вапніка [4], і вже є найбільш ефективним інструментом в завданні класифікації. Цей метод має дуже хорошу здатність до узагальнення. На відміну від проблеми навчання класичних нейронних мереж, де мінімізована функція помилки нелінійна по відношенню до оптімізі ванням змінним безлічі

потенційних мінімумів, МОВ призводить до квадратичного програмування з лінійними залежностями і може визначити чітко виражений глобальний мінімум. По суті, МОВ - це лінійний механізм, що працює в просторі характери стик з високою роздільною здатністю, створеному шляхом нелінійного перетворення вихідного *N* мірного вхідного вектора *x* в *K* -мірним простір характеристик (*K*>*N*) з використанням функції *φ (x)*. Рівняння гіперплощини, що розділяє два різних класи в *N* –мірному просторі:

$$y(x) = w^T \varphi(x) + \omega_0 = \sum_{j=1}^{K} \omega_j \varphi_j(x) + \omega_0 ,$$

де $\varphi(x)$ – це ваговий вектор мережі, w= [ω_1, ω_2, ..., ω_K] ^{T}, а ω_0 це похибка. Коли виконується y(x)> 0, вхідний вектор x належить до одного класу, а коли y(x) < 0 - до іншого. Всі математичні операції в режимі навчання і тестування виконуються з використанням так званої керн-функції K(x_i, x), що задовольняє умовам Мерцер [15]. Керн-функція визначається як внутрішнє скалярний твір вектора *φ (x)*, *K (x) = $\varphi^T(x_i)$ φ(x)*. Найбільш відомі керн-функції це лінійна, Гауссіан, поліноміальна і сплайн-функція. Основним завданням навчання в МОВ є поділ тренувальних векторів x_i на 2 класу, що визначаються приймають значеннями d_i=1 або d_i=-1, з максимальною роздільною здатністю. Далі виникає завдання максимізації квадратичної функції *Q (x)* , описаної в [15, 17, 18, 19]:

$$Q(\alpha) = \sum_{i=1}^{p} \alpha_i - 0.5 \sum_{i=1}^{p} \sum_{j=1}^{p} \alpha_i \alpha_j d_i d_j K(x_i, x_j)$$

Змінні α_i це коефіцієнти Лагранжа, d_i належать до приймаючих значень, відповідним вхідним векторам x_i, *C*- обумовлена користувачем константа

регуляризації, а p це кількість пар учнів даних (x_i, d_i). Вирішення цих двох завдань щодо коефіцієнтів Лагранжа дозволяє визначити оптимальний ваговий вектор w_{opt} мережі МОВ

$$w_{opt} = \sum_{i=1}^{N_{sv}} \alpha_i d_i \varphi(x_i),$$

де N_{sv} це кількість опорних векторів (векторів x_i, для яких коефіцієнти Лагранжа відмінні від нуля). Підставивши рішення (3) в рівняння (1), отримаємо вираз вихідного сигналу $y(x)$ мережі МОВ як функції кернів

$$y(x) = \sum_{i=1}^{N_{sv}} \alpha_i d_i K(x_i, x) + \omega_0$$

Позитивне значення $y(x)$ відповідає 1 (об'єкт належить до досліджуваного класу), а негативне значення -1 (об'єкт належить протилежному класу).

Основний параметр МОВ це константа регуляризації C.Вона контролює співвідношення між шириною роздільного кордону, що впливає на складність методу, і нерозділеними точками на етапі навчання мережі. Маленьке значення C призводить до розширення роздільного кордону і збільшення кількості нерозділених точок у фазі навчання. Велике значення C дозволяє досягти мінімальної кількості помилок класифікації, вузького роздільного кордону і меншого числа опорних векторів. Занадто велике значення C призводить до втрати узагальнюючої здатності навченою мережі. Для нормалізованих вхідних сигналів зі значеннями в інтервалі (-1, 1) константа регуляризації зазвичай суттєво більше 1 і визначається емпірично шляхом використання зразкового набору даних.

Важливим досягненням МОВ є перетворення завдання навчання в задачу квадратичного програмування. Для такого завдання оптимізації існує безліч ефективних алгоритмів навчання [19-21], які майже у всіх випадках призводять до глобального мінімуму цільової функції і вибору найкращого набору параметрів мережі. Іншим досягненням МОВ є хороша узагальнююча здатність, яка майже не залежить від кількості тренувальних зразків.

Хоча МОВ розділяє дані на 2 класу, розпізнати більшу кількість класів також нескладно шляхом застосування або методу «один проти одного» або методу «один проти всіх» [21]. Мето д «один проти одного» зазвичай більш ефективний. У цьому методі незалежно навчається безліч локальних двухклассових класифікаторів, що дозволяє вибрати найкращий класифікатор. Для *M* класів необхідно навчити *M(M-1)/ 2* двокласових, заснованих на МОВ системи розпізнавання.

2.5.6 Самоорганізовані карти

Нейронні мережі у вигляді самоорганізованих карт [30], представлені Конохеном, є одними з найбільш відомих неконтрольованих нейронних мереж. Вони визначають перехід з *M*-мірного вхідного простору в вихідний простір нижчого порядку (зазвичай одно-або двовимірне), в якому зберігаються вихідні ключові зв'язку між образами. Така модель характеризується скупченням вхідних образів у вигляді топографічної карти, на якій просторове розташування нейронів відображає схожість між групами образів. Позначення x_1-x_7 це набір кольорових образів, представлених у вигляді червоного, зеленого і синього значень. Після першого тренування СОК здатна згрупувати схожі образи. Червоні образи, наприклад, згруповані в верхньому правому куті карти (рисунок 2.11).

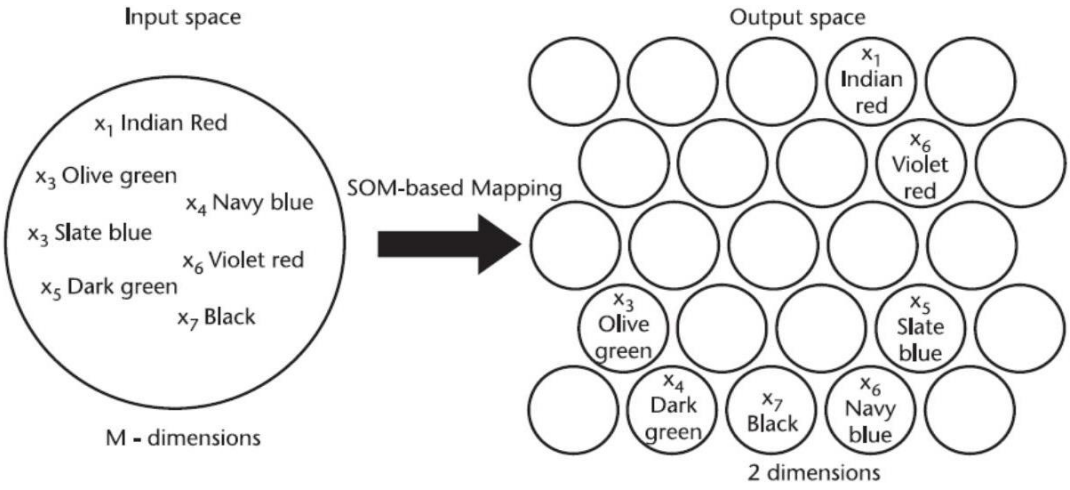

Рисунок 2.11 - Ілюстрація самоорганізованої карти

Традиційний алгоритм СОК. Відносна простота цього методу є його ключовою перевагою. Він не передбачає визначення цільової функції, тому не потрібні такі складні математичні операції як взяття похідних або обернення матриць.

У порівнянні з незмінною структурою ієрархічної кластеризації і недоліком структури в методі *k*-середніх, СОК відображають відносини подібності між образами і кластерами шляхом адаптування їх нейронів, які використовуються для опису образів-прототипів [31]. Однак, таке визначення топології карти призводить до неможливості автоматично визначати межі кластерів.

Існує безліч методів для розширення можливостей візуалізації даних в СОК. Один з них це побудова матриці відстаней (U-матриці), яка описує відстані між сусідніми нейронами і зображується на карті у вигляді колірної схеми (рисунок 2.12).

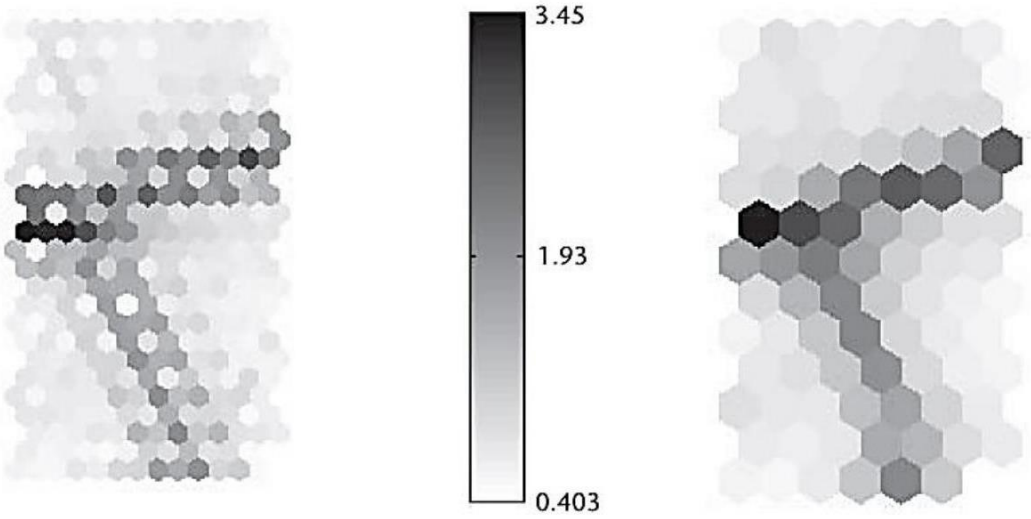

Рисунок 2.12 - Зображення даних для ірисів Фішера за допомогою СОК

U-матриця і карта, заснована на матриці середніх відстаней, представлені на рисунку 1.16 праворуч і ліворуч відповідно. Шестикутники це нейрони. Темні ділянки між нейронами відповідають великій відстані, а світлі говорять про те, що вхідні образи розташовані близько один до одного у вхідному просторі. Таким чином, світлі області можна прийняти за кластери, а темні – за їх межі. В даному випадку СОК виділила 3 кластери в наборі даних.

2.6 Висновки

Навідміну від методів навчання з учителем, метою традиційних методів навчання без вчителя є пошук відповідних кластерів, категорій або асоціацій за умови відсутності будь-якої попередньої інформації про класах в процесі навчання. У разі класифікації ЕКГ методи навчання без учителя застосовуються для вирішення різних завдань, в тому числі класифікації аритмій, розпізнавання і візуалізації образів.

Метод PCOK, представлений в підрозділі, має переваги перед традиційними моделями в області виявлення і візуалізації кластерів. Необхідно відзначити, що методи навчання без вчителя можуть бути адаптовані до застосування в задачах, що вирішуються за допомогою методів навчання з учителем.

РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ РОБОТИ НЕЙРОННИХ МЕРЕЖ

3.1 Опис вихідних даних

В якості вхідних даних використовувалося 6 записів ЕКГ, взятих з бази Фізіонет (The MGH / MF Waveform Database), а саме «mgh011», «Mgh022», «mgh031», «mgh102», «mgh108», «mgh121». Кожен сигнал містить від 1 до 12 відведень, тривалість сигналів становить 1 годину 40 хвилин (100 хв, 6000 с). Для тестування алгоритму вибиралися тільки ті сигнали, в яких чітко виділяються кілька типів ритмів. Таким чином, експеримент зводиться до багатокласової задачі. Оскільки у 2 главі було з'ясовано, що обидва типи нейронних мереж дають однакові результати, але швидкодія мережі розпізнавання образів вище, для тестування програмного комплексу була обрана мережа «patternnet». Вибір кількості прихованих шарів обумовлений тим, що в подальшому завдання класифікації може ускладнитися появою більш ніж двох класів комплексів. З теорії нейронних мереж відомо, що 1 прихований шар моделює лине йний роздільник, однак, якщо кількість класів більше, ніж два, для їх точної класифікації необхідна нелінійна розділяє функція, яка може бути змодельована тільки мережею з двома і більше прихованими шарами.

3.2 Попередня обробка сигналів

Попередня обробка сигналів потребує фільтрації сигналів, спочатку фільтром нижніх частот з частотою 1 Гц, потім режекторним фільтром з частотою 60 Гц для придушення мережної перешкоди. Після цього сигнал був

розділений на відведення. Після цього важливо було виявити QRS-комплекси для подальших розрахунків. Виявлення комплексів проводилося за допомогою функції findpeaks, яка повертає номер відліку сигналу, відповідного R-зубця. Після отримання точок всіх R-зубців, можна переходити і починати визначення всього комплексу.

У зв'язку із великими обсягами даних, було вирішено виділити в кожному з них тренувальний і тестовий шматки, тривалістю трохи більше півтора хвилин, що містять 171 QRS-комплекс, і використовувати отримані сигнали для верифікування якості алгоритму. Крім того, для кожного сигналу була проведена верифікація, аналогічно п. 2.2. Приклад тренувального і тестового шматків для сигналу «mgh121» зображений на рисунку 3.1.

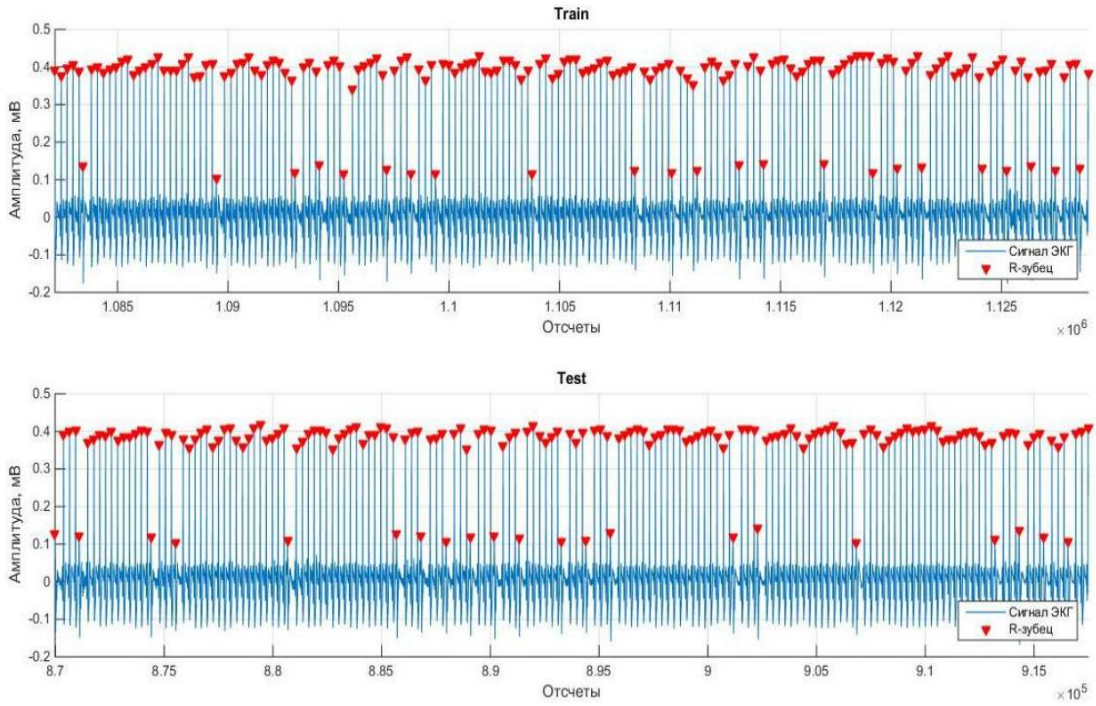

Рисунок 3.1- Тренувальний і тестовий шматки сигналу «mgh 121»

З рисунка неважко помітити, що в сигналі чітко виділяються 2 типи комплексів.

3.3 Вибір ознак для класифікації

В якості ознак класифікації були обрані елементи спектральної щільності потужності, розраховані з кроком 4 Гц в проміжку від 1 до 40 Гц. Вибір частотного діапазону обумовлений тим, що на частотах вище 40 Гц СПМ прямує до нуля. Таким чином, було отримано 10 ознак класифікації для кожного сигналу. Щоб класифікація була максимально точною і ефективною, необхідно з наявних 10 ознак відібрати тільки найбільш інформативні, для цього необхідно розрахувати мінімальне, максимально і середнє значення ознак для кожного класу і кожного відведення . Ознаки, для яких мінімальне і максимальне значення сильно відрізняються від середнього, не є інформативними і не будуть використовуватися для класифікації. Для простоти вибору ознак були побудовані гістограми, що містять максимальну, мінімально е і середнє значення кожної ознаки для двох класів в трьох відведеннях. Для наочності для кожного сигналу були побудовані графіки QRS-комплексів кожного класу у всіх відведеннях і відповідні їм графіки СПМ, а також усереднені спектри комплексів про боіх класів в трьох відведеннях. Ці графіки дозволяють візуально визначити відмінності між класифікуються комплексами і їх спектральними ознаками. Після вибору інформативних ознак можна приступати до навчання і тестування нейронної мережі. Результати класифікації будуть представлені у вигляді матриці помилок.

Графік тренувального і тестового шматків, QRS-комплексів і відповідних їм графіків СПМ, усереднених спектрів, гістограм розподілу ознак і матриці помилок для тренувального і т естового шматків для сигналу «mgh011» представлені на рисунках 3.2-3.7 відповідно.

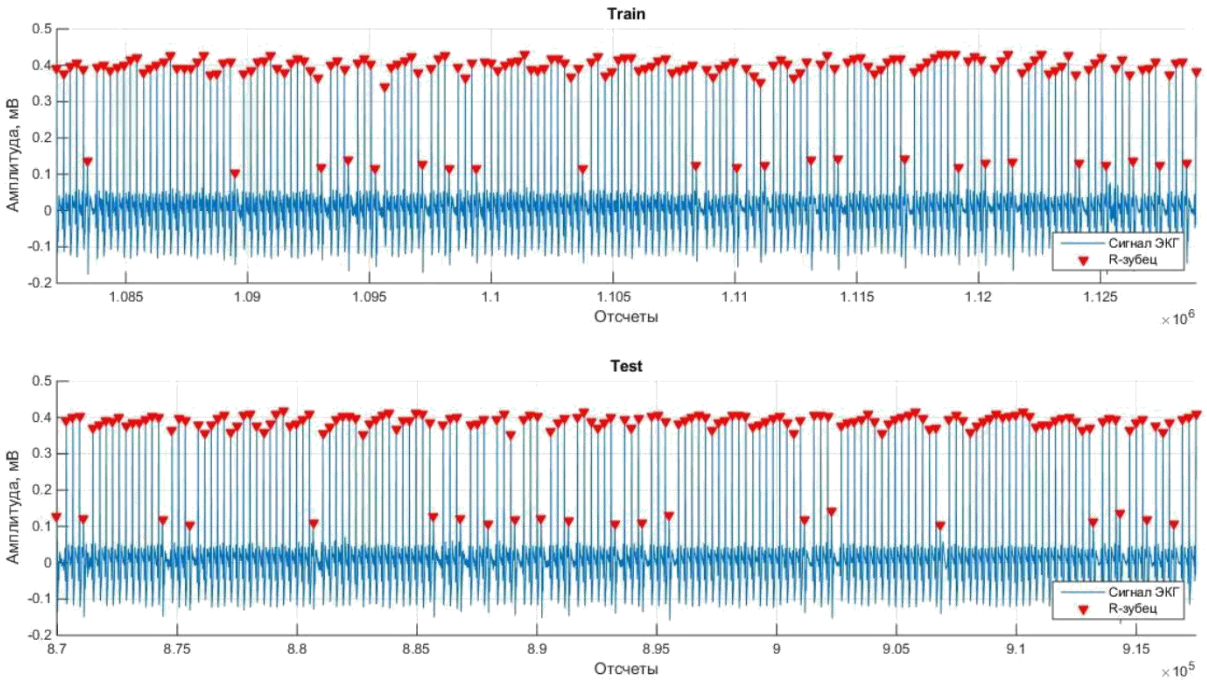

Рисунок 3.2 - Тренувальний і тестовий шматки для сигналу «mgh011»

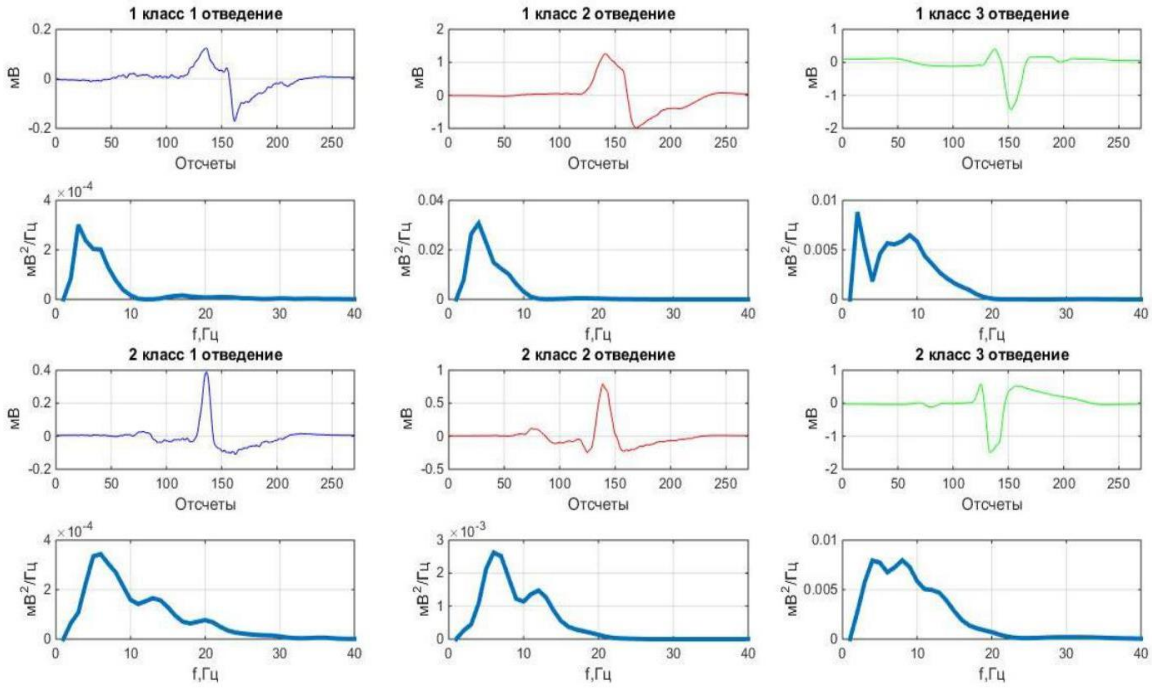

Рисунок 3.3 - QRS-комплекси і відповідні їм графіки СПМ для сигналу «mgh011»

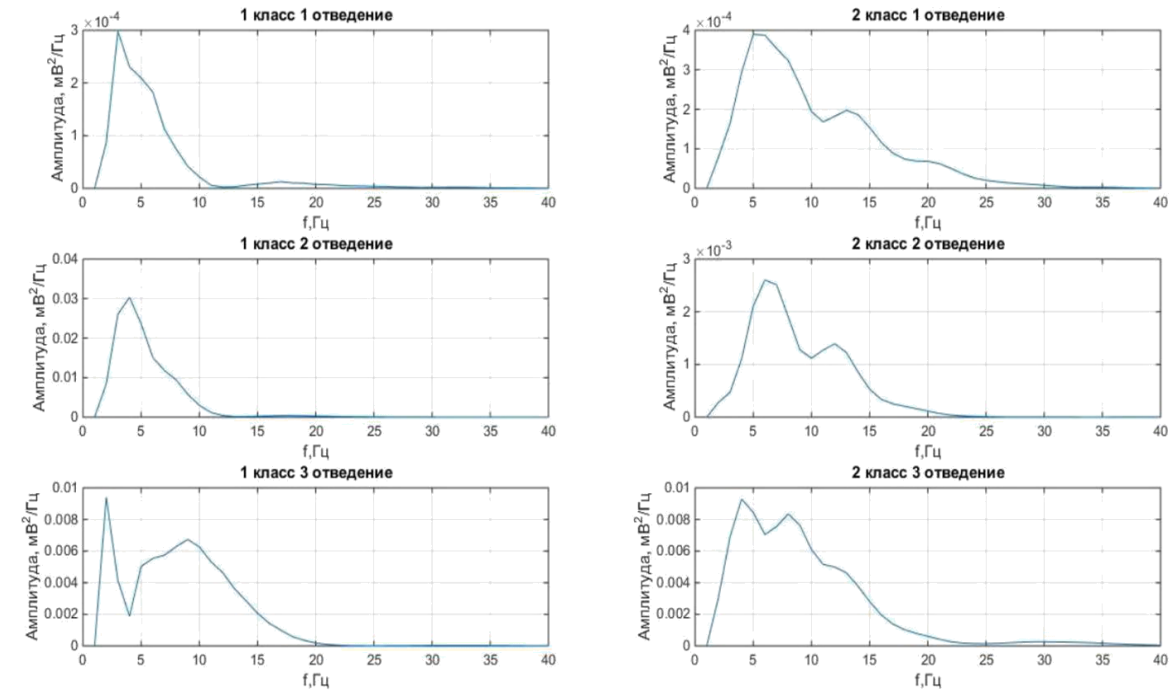

Рисунок 3.4 - Усереднені спектри для сигналу «mgh011»

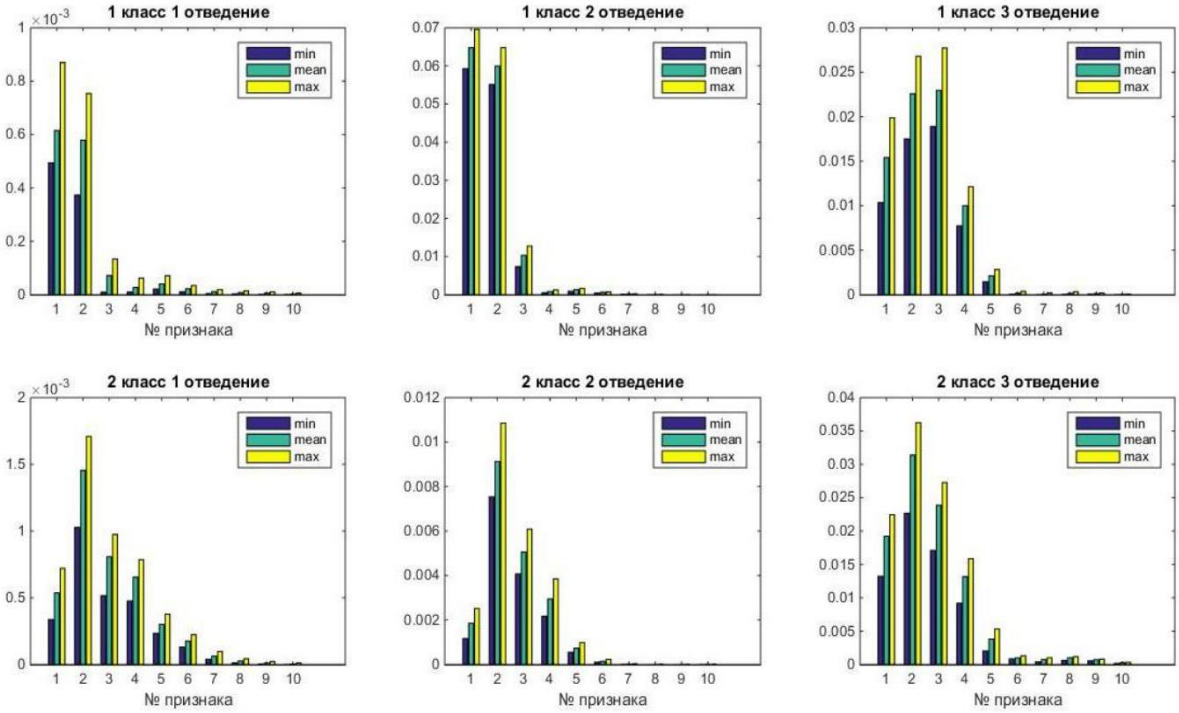

Рисунок 3.5 - Гістограми розподілу ознак для сигналу «mgh011»

Виходячи з видів гістограм розподілу ознак всіх сигналів видно, що інформативними можна вважати ознаки з 1 по 8, значить, вони будуть використовуватися як вхід нейронної мережі. Матриці помилок для всіх сигналів підтверджують припущення про інформативності вибраних ознак, так як мережа безпомилково класифікувала комплекси.

3.4 Оцінка якості роботи алгоритмів.

Для порівняння взяли 8 різних методів класифікації, для кожного з яких експериментальним шляхом обиралися найкращі параметри та сімейства вейвлетів. Результати наведені на рисунку 3.6.

Метод	Accuracy
Decision Tree	63-90%
Gradient Boosting	75-97%
Logistic Regression	69-94%
KNN	71-88%
Random Forest	72-94%
RNN	88-94%
CNN	92-99%
MLP	85-90%

Рисунок 3.6 – Результати порівняння методів

3.5 Висновки

У даній роботі необхідно було перевірити припущення про можливість класифікації форм ЕКГ, використовуючи в якості ознак класифікації

компоненти СПМ. Оцінка якості роботи створеного алгоритму повинна бути проведена шляхом порівняння його результатів з результатами вже наявних алгоритмів і досліджень. Однак, більшість відомих на даний момент алгоритмів класифікації мають на увазі, що в результаті буде отримано віднесення комплексів до конкретного патологічного класу (екстрасистоли, тахікардії і т.п.), тобто в них проведена дуже чітка верифікація. У даній же роботі верифікація являє собою віднесення комплексу до «нормі» або «патології» в загальному сенсі, тобто мова йде про формальну кластеризації. У проведеному дослідженні завдання класифікації зводиться до двухклассовой, однак, в подальшому може з'являтися більше класів, може знадобитися більш точна верифікація. Таким чином, буде необхідний перехід до більш складній структурі нейронної мережі, а їм енно більшій кількості прихованих шарів. Всі ці проблеми можуть бути завданнями для подальшого вдосконалення алгоритму.

Оскільки в роботі експеримент проводився на тренувальних і тестових шматках 6 сигналів, тобто на 12 шматках тривалістю близько 1,5 хвилин кожен, оцінити якість роботи алгоритму можна розрахувавши для них середню помилку класифікації. З матриць помилок в п. 3.3 видно, що для всіх сигналів загальна помилка класифікації дорівнює нулю, таким чином, середня помилка класифікації для 6 сигналів дорівнює нулю.

$$er = \frac{1}{n}\sum_{i=1}^{n} er_i = \frac{1}{12}\sum_{i=1}^{12} 12 * 0\% = 0\%$$

ВИСНОВКИ

Методи обробки сигналів, засновані на штучних нейронних мережах, в даний час знайшли широке застосування в клінічній практиці. Ці методи найбільш швидко розвиваються, навідміну від інших відомих методів класифікації форм ЕКГ, а також показують найкращі результати.

В ході виконання випускної кваліфікаційної роботи було зроблено припущення про те, що спектральна щільність потужності QRS-комплексу є досить інформативним показником і може використовуватися як ознака класифікації.

Метою даної роботи була розробка алгоритму автоматичної класифікації форм ЕКГ з використанням штучних нейронних мереж.

Для реалізації алгоритму використовувалася середовище програмування Matlab, мови програмування Python та R. Розроблений програмний комплекс дозволяє провести попередню обробку сигналів ЕКГ, розрахувати ознаки класифікації у вигляді спектральної щільності потужності, а також безпосередньо провести классиф ікацію форм QRS-комплексів за допомогою заздалегідь створених нейронних мереж.

В результаті проведеної роботи були отримані докази можливості класифікації форм ЕКГ на основі компонентів спектральної щільності потужності, при цьому точність класифікації зі ставила 100%.

Цінність розробленого програмного комплексу полягає в можливості його використання для класифікації великої кількості даних з використанням мінімального набору ознак.

РОЗДІЛ 4 СТАРТАП АНАЛІЗ ПРОЕКТУ

4.1 Інформаційна карта проекту

Таблиця 4.1 – Інформаційна карта

1. Назва проекту	Комплексна система класифікації кардіограм
2. Автори проекту	Мнухіна Катерина
3. Коротка анотація (не більше 1/3 сторінки)	Дана система призначена для класифікації кардіограм на класи «здоровий» і «хворий». На основі наявної бази даних, система визначає, чи є заданий сигнал кардіограмою людини. На наступному кроці, якщо сигнал є кардіограмою людини, система визначає його відповідність класам «здоровий» чи «хворий» і надсилає користувачу інформацію про стан здоров’я та необхідність відвідування лікаря
4. Термін реалізації проекту	6 місяців
5. Необхідні ресурси	Обладнання – компьютери, датчики, «розумні» браслети, мобільні телефони. Програмне забезпечення, операційна система, антивірусне обладнання. Електрика, Інтернет. Фінансові ресурси – заробітна плата розробникам і тестувальникам на 6 місяців роботи, гроші на на оплату аренди приміщення для роботи, аренди серверів, реклами тощо.

Продовження таблиці 4.1

6. Опис проблеми, яку вирішує проект	Дана комплексна система дозволяє розвязати проблему відслідковування хвороб серця у населення. На основі результатів класифікації відбувається розв'язання цієї проблеми – вчасне визначення хвороби на початковій стадії та попередження користувача про необхідність лікування.
7. Головні цілі та завдання проекту	Основна мета проекту – отримання диплому автором цього проекту. Додаткові завдання – новий досвід, розробка комплексної системи, робота із реальними даними та створення комерційно успішного продукту.

4.2. Технологічний аудит ідеї проекту

Даний розділ описує економічну обгрунтованість імплементації цього проекту. В табл. 4.2 приведено опис ідеї стартап-проекту.

Таблиця 4.2 – Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Дана комплексна система дозволяє розвязати проблему відслідковування хвороб серця у населення.	Розпізнавання пульсу людини	Можливість отримувати більш чистий сигнал

Продовження таблиці 4.2

Зміст ідеї	Напрямки застосування	Вигоди для користувача
На основі результатів класифікації відбувається розв'язання цієї проблеми – вчасне визначення хвороби на початковій стадії та попередження користувача про необхідність лікування.	2. Визначення стану здоров'я людини та наявності серцевих захворювань	Визначення захворювання на початковій стадії, вчасне лікування
	3. Аналіз стану здоров'я людини в динаміці	Можливість відстежування стану пацієнта

Проведено аналіз технологічної здійсненності проекту. Для реалізації основних частин проекту були підібрані відповідні технології. Результати аналізу наведено в таблиці 4.3.

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*				*(слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Конкурент 1*	*Конкурент 2*	*Конкурент 3*			
.	Точність прогнозування	Застосування кращої моделі	Власний алгоритм	Відсутнє прогнозування	Алгоритм *Lokad*			+

Продовження таблиці 4.3

п/п	*Техніко-економічні характеристики ідеї*	*(потенційні) товари/концепції конкурентів*				*(слабка сторона)*	*N (нейтральна сторона)*	*S (сильна сторона)*
		Мій проект	*Конкурент 1*	*Конкурент 2*	*Конкурент 3*			
.	Складнання розкладу операторів	Наявне складання приблизного розкладу	Дана функція повністю відсутня	Дана функція повністю відсутня	Дана функція повністю відсутня			+
.	Ризики невірного прогнозу	Існують, через велику кількість факторів.	Невідомо	Відсутні через відсутність прогнозу	Великий ризик невірного прогнозу		+	
.	Доступність, зручність	Графічний інтерфейс на основі Python	Власний інтерфейс	Власний інтерфейс	Власний інтерфейс		+	

Таблиця 4.4 – Технологічна здійсненність ідеї проекту

№ п/п	*Ідея проекту*	*Технології її реалізації*	*Наявність технологій*	*Доступність технологій*
1	Створення системи класифікації кардіограм	Використання мови програмування Python	Наявна	Доступні
2		Tensorflow	Наявна	Доступні
3		Використання мови програмування R	Наявна	Доступні
Обрана технологія реалізації ідеї проекту: мова програмування Python. Бібліотека Tensorflow				

4.3 Аналіз ринкових можливостей

Перед реалізацією проекта необхідно спланувати напрямки його розвитку враховуючи стан ринкового середовища, потреби споживачів та пропозиції конкурентів. Для цього необхідно визначити ринкові можливості, що можуть бути використані під час ринкового вровадження проекту, а також ринкові загрози, що можуть бути бути перешкодами реалізації проекту.

Рентабельність — поняття, що характеризує економічну ефективність виробництва, за якої за рахунок грошової виручки від реалізації продукції (робіт, послуг) повністю відшкодовує витрати на її виробництво й одержується прибуток як го. Значення середньої рентабельності визначено за

формулою:

$$R = \frac{P}{I \times n} * 100\%$$

де P - прибуток за час експлуатації проекту; I - повна сума інвестиційних витрат; n - час експлуатації проекту.

В таблиці 4.5 наведено результати попередньої характеристики ринку стартап-проекту.

Таблиця 4.5 – Попередня характеристика потенційного ринку стартап-проекту

п/п	*Показники стану ринку (найменування)*	*Характеристика*
	Кількість головних гравців, од	3
	Загальний обсяг продаж, грн/ум.од	100000
	Динаміка ринку (якісна оцінка)	Зростає
	Наявність обмежень для входу (вказати характер обмежень)	Немає
	Специфічні вимоги до стандартизації та сертифікації	Немає
	Середня норма рентабельності в галузі (або по ринку), %	27%

Інвестувати грошові засоби доцільно тоді, коли від цього можна отримати більший прибуток, ніж від їх зберігання у банку. Порівнюючи середньорічну рентабельність інвестицій зі ставкою банківського відсотка, можна дійти висновку, що вигідніше.

В таблиці 4.6 наведена характеристика потенційних клієнтів стартап-

проекту.

Таблиця 4.6 – Характеристика потенційних клієнтів стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
1	Потреба в відокремленні сигналу від шуму	Аналітики, аналітичні відділи, лікарі	Велика кількість даних.	Простота використання, висока точність
2	Створення якісного прогнозу та визначення наявності захворювання	Користувачі додатку	Цікавить простота у використанні, низька ціна підтримки системи	Швидкість обробки, низька ціна
3	Створення точного довгостроково го прогнозу, відстеження стану здоров’я користувача в часі	Великі медичні компанії, користувачі додатку	Цікавить передусім точність довгострокови х прогнозів	Висока якість прогнозу

Наступним кроком є аналіз факторів загроз, що має допомогти передбачити перешкоди на шляху реалізації та впровадження проекту.

Фактори загроз наведені в таблиці 4.7, фактори можливостей – у таблиці 4.8.

Таблиця 4.7 – Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Конкуренція	Цього року очікується вихід на ринок крупної іноземної компанії-конкурента	Пришвидшити вихід програмного продукту
2	Брак даних для навчання	Ускладнення прогнозів через недостатню кількість даних для навчання моделей	Укладання контрактів із медичними центрами

Таблиця 4.8 – Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	Гнучкі ціни	Зменшення ціни задля збільшення попиту	Введення власних гнучких цін
2	Диференціація витрат	Зменшення витрат за рахунок їх перерозподілу	Зменшення витрат на додаткові, непрофільні задачі.
3	Доступність	Адаптація системи для різних девайсів	Розробка ПЗ для мобільних телефонів та фітнес-браслетів

Наступним кроком є ступеневий аналіз конкуренції на ринку. Результати аналізу приведені в таблиці 4.9.

Таблиця 4.9 – Ступеневий аналіз конкуренції на ринку.

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
1. Вказати тип конкуренції - Досконала конкуренція	Багато систем/команд аналітиків	Розробити впізнаваний продукт, якість, що вирізнятиме нас від конкурентів
2. За рівнем конкурентної боротьби: міжнародний	На ринку присутні системи, розроблені за кордоном.	Розширення аудиторії, розширення списку мов, яві підтримуються системою
3. За галузевою ознакою - внутрішньогалузева	Конкуренти мають ПЗ, яке використовується лише всередині даної галузі	Розширення функціоналу для вирішення нових задач аналізу кардіограм
4. Конкуренція за видами товарів: товарно-видова	Види товарів є однаковими, а саме – програмне забезпечення для класифікації кардіосигналів.	Розробити продукт з урахуванням недоліків конкурентів.

Продовження таблиці 4.9

Особливості конкурентного середовища	*В чому проявляється дана характеристика*	*Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)*
5. За характером конкурентних переваг: Нецінова	Різні способи прогнозування дають різну точність	Розробка кращих(точніших) алгоритмів
6. За інтенсивністю: немарочна	Бренди відсутні	-

Після аналізу конкуренції проводиться більш детальний аналіз умов конкуренції в галузі. *М. Портер* вирізняє п‘ять основних факторів (рисунок 4.1), що впливають на привабливість вибору ринку з огляду на характер конкуренції.

Рисунок 4.1 – Складові моделі 5 сил М. Портера

Сильні позиції компанії за кожним з факторів означають її можливості забезпечити необхідні темпи обороту капіталу та її здатність впливати на інших агентів ринку, диктуючі їм власні умови співпраці.

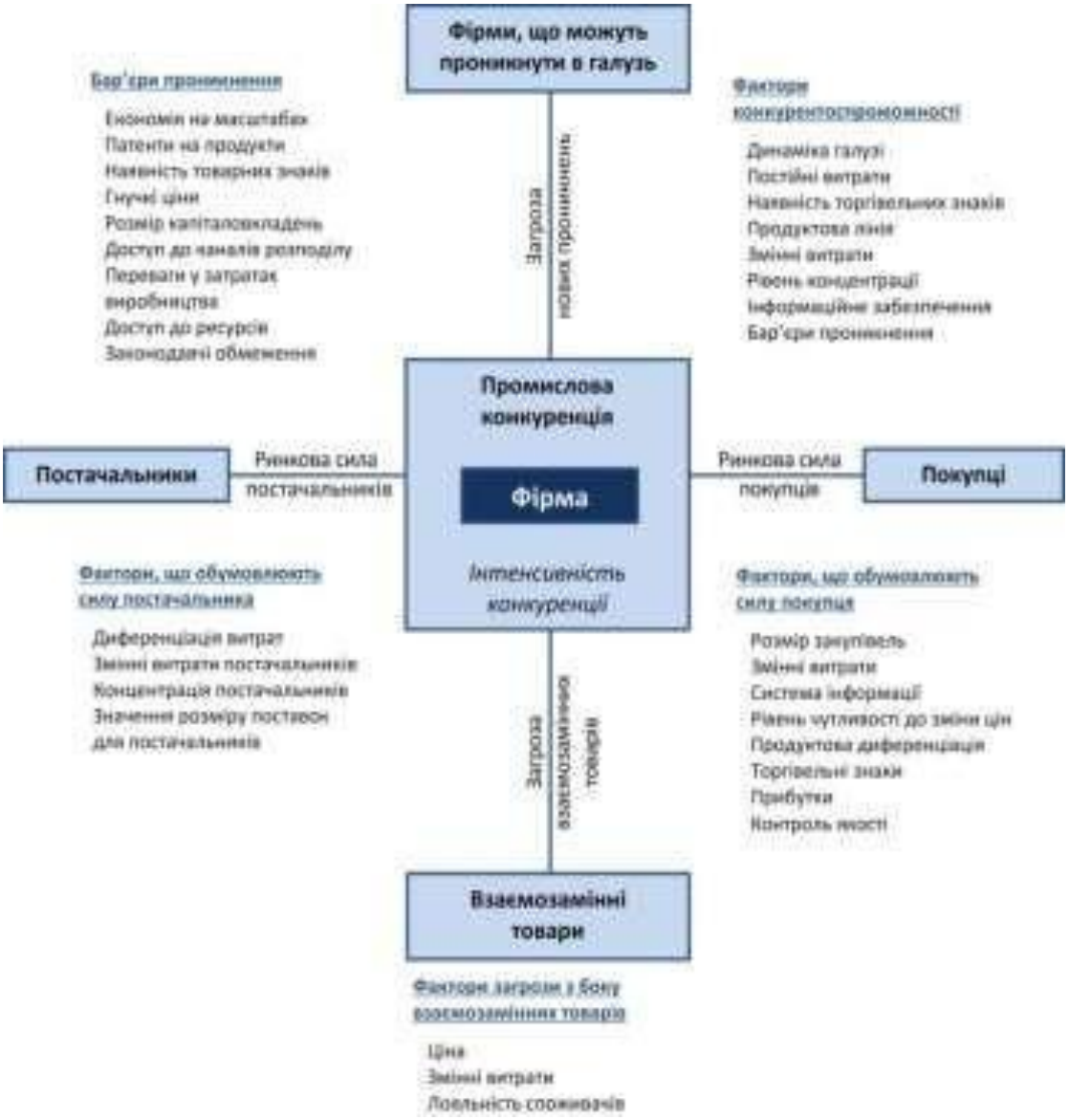

Рисунок 4.2 – Модель 5 сил М. Портера для аналізу конкуренції в галузі.

Характеристики факторів моделі відрізняються для різних галузей та змінюються із часом. Сила кожного фактору є функцією від структури галузі та її техніко-економічних характеристик. На основі аналізу складових моделі 5 сил М. Портера (рисунок 4.2) розробляється перелік факторів конкурентоспроможності для певного ринку зображенному в таблиці 4.10.

Таблиця 4.10 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	*Прямі конкуренти в галузі*	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
	Інші комплексні системи	*Гнучкі ціни, розмір капіталовкладень*	*Змінні витрати постачальників, диференціація витрат*	*Контроль якості, система інформації*	*Ціна, лояльність споживачів*
Висновки:	Інтенсивна конкуренція	Є як можливості входження на ринок, так і нові потенційні конкуренти	Постачальники не диктують умови роботи на ринку	Клієнти не диктують умови роботи на ринку	Обмеження для роботи на ринку через товари замінники

За результатами аналізу таблиці робиться висновок щодо принципової можливості роботи на ринку з огляду на конкурентну ситуацію. Також робиться висновок щодо характеристик (сильних сторін), які повинен мати проект, щоб бути конкурентоспроможним на ринку. Другий висновок враховується при формулюванні переліку факторів конкурентоспроможності.

На основі аналізу конкуренції, а також із урахуванням характеристик ідеї проекту, вимог споживачів до товару та факторів маркетингового

середовища визначається та обґрунтовується перелік факторів конкурентоспроможності. Враховуючи отриманні результати аналіз оформлюється за табл. 4.11.

Таблиця 4.11 – Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1	Багатофункціональність	Жоден конкурент не є настільки багатункціональним, не здатен на прогноз, чистку шумів та класифікацію сигналів.
2	Якість	Висока якість прогнозу, велика кількість допоміжних статистичних даних
3	Підтримка клієнтів	Робота з клієнтами – великими компаніями та окремими спеціалістами

За визначеними факторами конкурентоспроможності проводиться аналіз сильних та слабких сторін стартап-проекту. Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) на основі виділених ринкових загроз та можливостей, та сильних і слабких сторін.

У таблиці 4.12 описано порівняльний аналіз сильних та слабких сторін проекту.

Таблиця 4.12 – Порівняльний аналіз сильних та слабких сторін проекту

п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів*						
			-3	*-2*	*-1*	*0*	*1*	*2*	*3*
1	Багатофункціональність	15		2	1		23		
2	Якість	20		12		22	22		
3	Підтримка клієнтів	5					22		

Перелік ринкових загроз та ринкових можливостей складається на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами) впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення. Наприклад: зниження доходів потенційних споживачів – фактор загрози, на основі якого можна зробити прогноз щодо посилення значущості цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

SWOT-аналіз стартап-проекту представленний у таблиці 4.13.

Таблиця 4.13 – SWOT-аналіз стартап-проекту

Сильні сторони: Висока якість прогнозу Якість Багатофункціональність	Слабкі сторони: Інтерфейс користувача Немає налагодженої партнерської бази
Можливості: Попит Зміна рівня доходів населення	Загрози: Конкуренція Брак даних для навчання

На основі SWOT-аналізу розробляються альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок. Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів. Альтернативи ринкового впровадження стартап-проекту представленні у таблиці 4.14.

Таблиця 4.14 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
1	Швидкий вихід на ринок із «сирим» продуктом, можливі проблеми із точністю прогнозу та доступністю на деяких платформах	35%	3 місяці
2	Поступовий вихід з готовим, відлагодженим продуктом. Висока якість, наявна база партнерів серед медичних центрів, доступність на більшості мобільних девайсів.	65%	6 місяців

4.3 Розробка ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів що відображається у таблиці 4.15.

Таблиця 4.15 – Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
1	Окремі аналітики	Низька готовність	25%	Висока	Середня
2	Малі та середні медичні центри, окремі лікарі	Висока	35%	Середня	Середня
3	Великі медичні компанії з великою базою даних	Висока	40%	Низька	Висока
Які цільові групи обрано:2,3					

За результатами аналізу потенційних груп споживачів (сегментів) автори ідеї обирають цільові групи, для яких вони пропонуватимуть свій товар, та визначають стратегію охоплення ринку.

Для роботи в обраних сегментах ринку необхідно сформувати базову стратегію розвитку. За М. Портером, існують три базові стратегії розвитку, що відрізняються за ступенем охоплення цільового ринку та типом конкурентної переваги, що має бути реалізована на ринку (за витратами або визначними якостями товару).

Стратегія лідерства по витратах передбачає, що компанія за рахунок чинників внутрішнього і/або зовнішнього середовища може забезпечити більшу, ніж у конкурентів маржу між собівартістю товару і середньоринковою ціною (або ж ціною головного конкурента).

Стратегія диференціації передбачає надання товару важливих з точки зору споживача відмітних властивостей, які роблять товар відмінним від товарів конкурентів. Така відмінність може базуватися на об‘єктивних або суб‘єктивних, відчутних і невідчутних властивостях товару (у ширшому розумінні – комплексі маркетингу), бути реальною або уявною. Інструментом реалізації стратегії диференціації є ринкове позиціонування.

Стратегія спеціалізації передбачає концентрацію на потребах одного цільового сегменту, без прагнення охопити увесь ринок. Мета тут полягає в задоволенні потреб вибраного цільового сегменту краще, ніж конкуренти. Мета тут полягає в задоволенні потреб вибраного цільового сегменту краще, ніж конкуренти. Така стратегія може спиратися як на диференціацію, так і на лідерство по витратах, або і на те, і на інше, але тільки у рамках цільового сегменту.

Проте низька ринкова доля у разі невдалої реалізації стратегії може істотно підірвати конкурентоспроможність компанії.

В таблиці 4.16 приведені деталі базової стратегії розвитку.

Таблиця 4.16 – Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
1	2 та 3	Стратегія диференційованого маркетингу	Висока універсальність, багатогалузевість, висока якість, ціна.	Стратегія лідерства по витратах

Наступним кроком є вибір стратегії конкурентної поведінки.

Стратегія лідера. Залежно від міри сформованості товарного (галузевого) ринку, характеру конкурентної боротьби компанії-лідери обирають одну з трьох стратегій: розширення первинного попиту, оборонну або наступальну стратегію або ж застосувати демаркетинг або диверсифікацію.

Стратегія розширення первинного попиту доцільна у разі, якщо фірмі-лідерові недоцільно розмінюватися на боротьбу з невеликими конкурентами, вона може отримати велику економічну віддачу від розширення первинного рівня попиту. В цьому випадку компанія займається реалізацією заходів по формуванню попиту (навчанню споживачів користуванню товаром, формування регулярного попиту, збільшення разового споживання), також пропаганду нових напрямів застосувань існуючих товарів, виявлень нових груп споживачів.

У міру зростання ринку, його становлення позиції компанії-новатора починають атакувати конкуренти-імітатори. В цьому випадку, компанія може

вибрати оборонну стратегію, метою якої є захист власної ринкової долі.

Наступальна стратегія припускає збільшення своєї частки ринку. При цьому переслідувана мета полягає в подальшому підвищенні прибутковості роботи компанії на ринку за рахунок максимального використання ефекту масштабу.

Якщо фірма потрапляє під дію антимонопольного законодавства, вона може удатися до стратегії демаркетинга, що припускає скорочення своєї частки ринку, зниження рівня попиту на деяких сегментах за рахунок підвищення ціни. При цьому ставиться завдання недопущення на ці сегменти конкурентів, а компенсація втрат прибутку через зменшення обсягів виробництва компенсується встановленням надвисоких цін.

Стратегія виклику лідера. Стратегію виклику лідерові найчастіше вибирають компанії, які є другими, третіми на ринку, але бажають стати лідером ринку. Теоретично, ці компанії можуть прийняти два стратегічні рішення: атакувати лідера у боротьбі за частку ринку або ж йти за лідером.

Рішення атакувати лідера є досить ризикованим. Для його реалізації потрібні значні фінансові витрати, know – how, краще співвідношення «ціна-якість», переваги в системі розподілу і просування і т. д. У разі не реалізації цієї стратегії, компанія може бути відкинута на аутсайдерські позиції на досить довгий час. Залежно від цього компанія може вибрати одну з альтернативних стратегій: фронтальної або флангової атаки.

Стратегія наслідування лідеру. Компанії, що приймають слідування за лідером – це підприємства з невеликою часткою ринку, які вибирають адаптивну лінію поведінки на ринку, усвідомлюють своє місце на нім і йдуть у фарватері фірм-лідерів. Головна перевага такої стратегії – економія фінансових ресурсів, пов‘язаних з необхідністю розширення товарного(галузевого) ринку, постійними інноваціями, витратами на утримання домінуючого положення.

Стратегія заняття конкурентної ніші. При прийнятті стратегії зайняття

конкурентної ніші (інші назви – стратегія фахівця або нішера) компанія в якості цільового ринку вибирає один або декілька ринкових сегментів. Головна особливість – малий розмір сегментів/сегменту. Ця конкурентна стратегія являється похідною від такої базової стратегії компанії, як концентрація. Головне завдання для компаній, що вибирають стратегію нішера або фахівця, – це постійна турбота про підтримку і розвиток своєї конкурентної переваги, формування лояльності і прихильності споживачів, підтримка вхідних бар‘єрів. Визначення базової стратегії конкурентної поведінки представленно у таблиці 4.17.

Таблиця 4.17 – Визначення базової стратегії конкурентної поведінки

№ п/п	*Чи є проект «першопрохідцем» на ринку?*	*Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?*	*Чи буде компанія копіювати основні характеристики товару конкурента, і які?*	*Стратегія конкурентної поведінки**
	Ні	Так	Ні	Зайняття конкурентної ніші

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розробляється стратегія

позиціонування, що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Визначення оптимальної стратегії позиціонування представленно у таблиці 4.17.

Таблиця 4.18 – Визначення стратегії позиціонування

п/п	*Вимоги до товару цільової аудиторії*	*Базова стратегія розвитку*	*Ключові конкурентоспроможні позиції власного стартап-проекту*	*Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)*
	Якість, точність, простота у використанні	Стратегія лідерства по витратах	Якість прогнозу, універсальність, велика кількість статистичної інформації, використання передових технологій та методів	Швидкість Точність Позиціювання на низькій ціні

4.4 Розробка маркетингової програми

Першим кроком є формування маркетингової концепції товару, який отримає споживач. Для цього у табл. 4.18 потрібно підсумувати результати попереднього аналізу конкурентоспроможності товару.

Надалі розробляється трирівнева маркетингова модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (табл. 4.20).

1-й рівень. При формуванні задуму товару вирішується питання щодо того, засобом вирішення якої потреби і / або проблеми буде даний товар, яка його основна вигода. Дане питання безпосередньо пов'язаний з формуванням технічного завдання в процесі розробки конструкторської документації на виріб.

Таблиця 4.19 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1	Висока швидкість класифікації кардіосигналів	Робота із використанням сучасних методів	Перевага у швидкості обробки даних та прийняття рішень
2	Створення якісного та точного висновку	Роботу із передовими архітектурами моделей,сучасних бібліотек	Розробка коротко- та довгострокових прогнозів, використання методів для покращення точності прогнозу

2-й рівень. Цей рівень являє рішення того, як буде реалізований товар в реальному/ включає в себе якість, властивості, дизайн, упаковку, ціну.

3-й рівень. Товар з підкріпленням (супроводом) - додаткові послуги та переваги для споживача, що створюються на основі товару за задумом і товару в реальному виконанні (гарантії якості, доставка, умови оплати та ін).

Після формування маркетингової моделі товару слід особливо відмітити – чим саме проект буде захищено від копіювання – за рахунок захисту ідеї товару (захист інтелектуальної власності), або ноу-хау, чи комплексне поєднання властивостей і характеристик, закладене на другому та третьому рівнях товару. Опис рівнів моделі товару представленний у таблиці 4.19, межі встановлення ціни – у таблиці 4.22.

Таблиця 4.20 – Опис трьох рівнів моделі товару

Рівні товару	*Сутність та складові*		
I. Товар за задумом	Комплексна система обслуговування кол-центрів		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Висока точність 2. Висока швидкість 3. Адаптованість системи на велику кількість девайсів		
	Якість – тестування сторонніми фірмами		
	Пакування – зручний інсталятор		
	Маркування відсутнє		
III. Товар із підкріпленням	Варіанти підписок для окремих користувачів та великих корпорацій, пробна демо-версія		
	Після продажу: підтримка користувачів		

Наступним кроком є визначення оптимальної системи збуту, в межах якого приймається рішення (табл. 4.20).

Таблиця 4.21 – Формування системи збуту

№	*Специфіка закупівельної поведінки цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1.	Базовий функціонал ПЗ для мобільних девайсів (телефонів, браслетів)	Поширення продукту	0 (напряму), 1 (через одного посередника)	Власна та через посередників
2	Отримання розширеного функціоналу для мобільних пристроїв	Продажа	0 (напряму), 1 (через одного посередника)	Власна та через посередників

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (табл. 4.21).

Таблиця 4.22 – Концепція маркетингових комунікацій

№	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
1.	Купівля ПЗ через інтернет, робота з ПЗ на різних типах пристроїв	Інтернет	Швидкодія, простота використання, Точність класифікації	Показати переваги ПЗ, у тому числі і перед конкурентами	Креативна реклама, що показує які можливості надає ПЗ

Таблиця 4.23 – Визначення меж встановлення ціни

№	*Рівень цін на товари-замінники*	*Рівень цін на товари аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послуг*
1.	5.5	-	2000000	-

4.5 Елементи фінансової підтримки стартапу та аналіз ризиків

У таблиці 4.23 наведено структуру інвестиційних витрат на реалізацію стартап-проекту.

Таблиця 4.23 – Сукупні інвестиційні витрати на реалізацію

№ з/п	***Стаття итрат***	***Сукупні витрати, тис. грн.***
1.	*Початкові витрати*	200
1.1.	Проведення пошукових та прикладних досліджень	75
1.2.	Розробка проектних матеріалів і ТЕО	25
1.3.	Проектування	20
1.4.	Придбання обладнання та устаткування	15

Таблиця 4.24 – Інвестиційні витрати

1.5.	Придбання ематеріальних активів	100
1.6.	Утримання обладнання та приміщень	30
1.7.	Попередні маркетингові дослідження	30
1.8.	Створення мережі збуту	20
1.9.	Просування та реклами	50
1.10	Юридичні послуги	15
2.	Матеріальні ресурси	0
2.1.	Матеріали	0
2.3.	Комплектуючі	0
2.4.	Сировина	0

3.	Оплату праці команди стартапу	175
	Разом:	755

4. 6 Висновки до розділу

Проведені дослідження показують існування можливості ринкової комерціалізації проекту, оскільки існують потенційні сегменти користувачів. Проте існує висока конкуренція на ринку, що потребує активного аналізу конкурентів та пошуку удосконалення існуючого функціоналу, та пошуку його альтернативного застосування.

Для успішного виконання проекту необхідно реалізувати програму із використанням засобів Tensorflow. В рамках даного дослідження були розраховані основні фінансово-економічні показники проекту, а також проведений менеджмент потенційних ризиків. Згідно з отриманими результатами економічного та рикового аналізу подальша імплементація є доцільною.

ВИСНОВКИ

Методи обробки сигналів, засновані на штучних нейронних мережах, в даний час знайшли широке застосування в клінічній практиці. Ці методи є найбільш швидко розвиваються, на відміну від інших відомих методів класифікації форм ЕКГ, а також показують найкращі результати.

В ході виконання випускної кваліфікаційної роботи було зроблено припущення про те, що спектральна щільність потужності QRS-комплексу яв ляется досить інформативним показником і може використовуватися як ознака класифікації.

Метою даної роботи була розробка алгоритму автоматичної класифікації форм ЕКГ з використанням штучних нейронних мереж.

Для реалізації алгоритму використовувалася середовище програмування Matlab. Розроблений програмний комплекс дозволяє провести попередню обробку сигналів ЕКГ, розрахувати ознаки класифікації у вигляді спектральної щільності потужності, а також безпосередньо провести классиф ікацію форм QRS-комплексів за допомогою заздалегідь створених нейронних мереж.

В результаті проведеної роботи були отримані докази можливості класифікації форм ЕКГ на основі компонентів спектральної щільності потужності, при цьому точність класифікації зі ставила 100%.

Цінність розробленого програмного комплексу полягає в можливості його використання для класифікації великої кількості даних з використанням мінімального набору ознак.

ПЕРЕЛІК ПОСИЛАНЬ

1. Р. М. Баевский, П. Я. Довгалевский, Ю. А. Кукушкин, М. М. Миронова, Анализ вариабельности сердечного ритма при использовании различных электрокардиографических систем. *Вестник аритмологии*. 2001, No.24. С. 65-87.

2. Eirini Christinaki, Giorgos Giannakakis, Franco Chiarugi. Comparison of Blind Source Separation Algorithms for Optical Heart Rate Monitoring. *Wireless Mobile Communication and Healthcare*. Milan, Italy. 2014, P. 339-342.

3. Дьяконов В., Абраменкова И. MATLAB. Обработка сигналов и изображений: Специальный справочник. СПб.: Питер, 2002. 608 с.

4. C. Takano, Y. Ohta. Heart rate measurement based on a time-lapse image. *Medical Engineering & Physics*. 2007. Vol.29, No. 8. P. 853-857.

5. Raducanu B., Kumar S. ECG analysis using wavelet transformation. *Computational Statistics & Data Analysis*. 2011. Vol.2, No. 1. P. 57-83.

6. Ghayoumi M. A quick review of deep learning in facial expression. *Journal of Communication and Computer*, 2017. Vol. 1. P. 34–38.

7. Kahou, S.E., Michalski V., Konda K. Recurrent neural networks for ECG classification. *The ACM on International Conference on Multimodal Interaction*, Seattle, WA, USA, 9–13 November 2015, P. 467–474.

8. Zhan C., Li W., Ogunbona P., Safaei F. A real-time cardiosignals recognition system for health care. *International Journal of Computer Technology*, 2008. Vol. 2, No.1. P. 10-20

9. Mehrabian, A. Communication without words. *Psychol. Today*. 1968, Vol. 2. P. 53–56.

10. Dornaika F., Raducanu B. Efficient ECG clustering for human robot interaction. *The 9th International Work-Conference on Artificial Neural Networks*

on Computational and Ambient Intelligence, San Sebas Xtián, Spain, 20–22 June 2007, P. 700–708.

11. Bartneck C., Lyons M.J. HCI and the heart: Towards an art of the soluble. *In Proceedings of the International Conference on Human-Computer Interaction: Interaction Design and Usability*, Beijing, China, 22–27 July 2007, P. 20–29.

12. Assari M.A., Rahmati M. Driver drowsiness detection using heart signals recognition. *The IEEE International Conference on Signal and Image Processing Applications*, Kuala Lumpur, Malaysia, 16–18 November 2011, P. 337–341.

13. Kumar S. ECG recognition: A review. *The National Conference on Cloud Computing and Big Data*, Shanghai, China, 4–6 November 2015, P 159–162.

14. Mourão A., Magalhães J. Competitive affective gaming: Winning with a smile. *The ACM International Conference on Multimedia*, Barcelona, Spain, 21–25 October 2013, P. 83–92.

15. Walecki R., Rudovic O. Deep structured learning for signal processing. *Image Vis. Comput*. 2017. Vol.1. P. 143–154

16. Kolakowaska A. A review of cardiosignals classification methods based on wavelet analysis. *The 6th International Conference on Human System Interaction*, Gdansk, Poland, 6–8 June 2013, P. 548–555

17. Anuati S.M., Mahoor M.H., Bartlett K., Trinh P., Cohn J. DISFA: A spontaneous heartbeat database. *IEEE Transactions on Affective Computing*, 2013, Vol.5. P. 151–160

18. HRB. URL: https://hrbheartdb.eu/ (дата звернення 22.09.2019)

19. KDEF. URL: http://www.emotionlab.se/resources/kdef (дата звернення 22.09.2019).

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

```
import os
import time
import numpy as np
import pandas as pd
import scipy.io as sio
from IPython.display import display

import matplotlib.pyplot as plt
import pywt
import scipy.stats

import datetime as dt
from collections import defaultdict, Counter

from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import AdaBoostClassifier
from sklearn.gaussian_process import GaussianProcessClassifier

    "Gradient Boosting Classifier": GradientBoostingClassifier(),
    "Random Forest": RandomForestClassifier(),
    "Logistic Regression": LogisticRegression(),
    "Nearest Neighbors": KNeighborsClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "Linear SVM": SVC(),
    "Neural Net": MLPClassifier(alpha = 1),
    "Naive Bayes": GaussianNB(),
    "AdaBoost": AdaBoostClassifier(),
    "Gaussian Process": GaussianProcessClassifier()
}

def batch_classify(X_train, Y_train, X_test, Y_test, no_classifiers = 5, verbose =
True):
    """
```

```
    This method, takes as input the X, Y matrices of the Train and Test set.
    And fits them on all of the Classifiers specified in the dict_classifier.
    Usually, the SVM, Random Forest and Gradient Boosting Classifier take quiet some time to train.
    So it is best to train them on a smaller dataset first and
    decide whether you want to comment them out or not based on the test accuracy score.
    """

    dict_models = {}
    for classifier_name, classifier in list(dict_classifiers.items())[:no_classifiers]:
        t_start = time.clock()
        classifier.fit(X_train, Y_train)
        t_end = time.clock()

        t_diff = t_end - t_start
        train_score = classifier.score(X_train, Y_train)
        test_score = classifier.score(X_test, Y_test)

        dict_models[classifier_name] = {'model': classifier, 'train_score': train_score, 'test_score': test_score, 'train_time': t_diff}
        if verbose:
            print("trained {c} in {f:.2f} s".format(c=classifier_name, f=t_diff))
    return dict_models

def get_train_test(df, y_col, x_cols, ratio):
    """
    This method transforms a dataframe into a train and test set, for this you need to specify:
    1. the ratio train : test (usually 0.7)
    2. the column with the Y_values
    """
    mask = np.random.rand(len(df)) < ratio
    df_train = df[mask]
    df_test = df[~mask]

    Y_train = df_train[y_col].values
    Y_test = df_test[y_col].values
    X_train = df_train[x_cols].values
    X_test = df_test[x_cols].values
    return df_train, df_test, X_train, Y_train, X_test, Y_test

def display_dict_models(dict_models, sort_by='test_score'):
```

```
    cls = [key for key in dict_models.keys()]
    test_s = [dict_models[key]['test_score'] for key in cls]
    training_s = [dict_models[key]['train_score'] for key in cls]
    training_t = [dict_models[key]['train_time'] for key in cls]

    df_ = pd.DataFrame(data=np.zeros(shape=(len(cls),4)), columns = ['classifier',
'train_score', 'test_score', 'train_time'])
    for ii in range(0,len(cls)):
        df_.loc[ii, 'classifier'] = cls[ii]
        df_.loc[ii, 'train_score'] = training_s[ii]
        df_.loc[ii, 'test_score'] = test_s[ii]
        df_.loc[ii, 'train_time'] = training_t[ii]

    display(df_.sort_values(by=sort_by, ascending=False))

def calculate_entropy(list_values):
    counter_values = Counter(list_values).most_common()
    probabilities = [elem[1]/len(list_values) for elem in counter_values]
    entropy=scipy.stats.entropy(probabilities)
    return entropy

def calculate_statistics(list_values):
    n5 = np.nanpercentile(list_values, 5)
    n25 = np.nanpercentile(list_values, 25)
    n75 = np.nanpercentile(list_values, 75)
    n95 = np.nanpercentile(list_values, 95)
    median = np.nanpercentile(list_values, 50)
    mean = np.nanmean(list_values)
    std = np.nanstd(list_values)
    var = np.nanvar(list_values)
    rms = np.nanmean(np.sqrt(list_values**2))
    return [n5, n25, n75, n95, median, mean, std, var, rms]

def calculate_crossings(list_values):
    zero_crossing_indices = np.nonzero(np.diff(np.array(list_values) > 0))[0]
    no_zero_crossings = len(zero_crossing_indices)
    mean_crossing_indices = np.nonzero(np.diff(np.array(list_values) >
np.nanmean(list_values)))[0]
    no_mean_crossings = len(mean_crossing_indices)
    return [no_zero_crossings, no_mean_crossings]

def get_features(list_values):
    entropy = calculate_entropy(list_values)
```

```
    crossings = calculate_crossings(list_values)
    statistics = calculate_statistics(list_values)
    return [entropy] + crossings + statistics

def get_uci_har_features(dataset, labels, waveletname):
    uci_har_features = []
    for signal_no in range(0, len(dataset)):
        features = []
        for signal_comp in range(0,dataset.shape[2]):
            signal = dataset[signal_no, :, signal_comp]
            list_coeff = pywt.wavedec(signal, waveletname)
            for coeff in list_coeff:
                features += get_features(coeff)
        uci_har_features.append(features)
    X = np.array(uci_har_features)
    Y = np.array(labels)
    return X, Y
activities_description = {
    1: 'walking',
    2: 'walking upstairs',
    3: 'walking downstairs',
    4: 'sitting',
    5: 'standing',
    6: 'laying'
}

def read_signals(filename):
    with open(filename, 'r') as fp:
        data = fp.read().splitlines()
        data = map(lambda x: x.rstrip().lstrip().split(), data)
        data = [list(map(float, line)) for line in data]
    return data

def read_labels(filename):
    with open(filename, 'r') as fp:
        activities = fp.read().splitlines()
        activities = list(map(int, activities))
    return activities

def randomize(dataset, labels):
    permutation = np.random.permutation(labels.shape[0])
    shuffled_dataset = dataset[permutation, :, :]
    shuffled_labels = labels[permutation]
```

```
    return shuffled_dataset, shuffled_labels

INPUT_FOLDER_TRAIN = './data/UCI_HAR/train/InertialSignals/'
INPUT_FOLDER_TEST = './data/UCI_HAR/test/InertialSignals/'

INPUT_FILES_TRAIN = ['body_acc_x_train.txt', 'body_acc_y_train.txt',
'body_acc_z_train.txt',
                     'body_gyro_x_train.txt', 'body_gyro_y_train.txt',
'body_gyro_z_train.txt',
                     'total_acc_x_train.txt', 'total_acc_y_train.txt', 'total_acc_z_train.txt']

INPUT_FILES_TEST = ['body_acc_x_test.txt', 'body_acc_y_test.txt',
'body_acc_z_test.txt',
                    'body_gyro_x_test.txt', 'body_gyro_y_test.txt', 'body_gyro_z_test.txt',
                    'total_acc_x_test.txt', 'total_acc_y_test.txt', 'total_acc_z_test.txt']

LABELFILE_TRAIN = './data/UCI_HAR/train/y_train.txt'
LABELFILE_TEST = './data/UCI_HAR/test/y_test.txt'

train_signals, test_signals = [], []

for input_file in INPUT_FILES_TRAIN:
    signal = read_signals(INPUT_FOLDER_TRAIN + input_file)
    train_signals.append(signal)
train_signals = np.transpose(np.array(train_signals), (1, 2, 0))

for input_file in INPUT_FILES_TEST:
    signal = read_signals(INPUT_FOLDER_TEST + input_file)
    test_signals.append(signal)
test_signals = np.transpose(np.array(test_signals), (1, 2, 0))

train_labels = read_labels(LABELFILE_TRAIN)
test_labels = read_labels(LABELFILE_TEST)

[no_signals_train, no_steps_train, no_components_train] = np.shape(train_signals)
[no_signals_test, no_steps_test, no_components_test] = np.shape(train_signals)
no_labels = len(np.unique(train_labels[:]))

print("The train dataset contains {} signals, each one of length {} and {}
components ".format(no_signals_train, no_steps_train, no_components_train))
print("The test dataset contains {} signals, each one of length {} and {}
components ".format(no_signals_test, no_steps_test, no_components_test))
```

```
print("The train dataset contains {} labels, with the following distribution:\n
{}".format(np.shape(train_labels)[0], Counter(train_labels[:])))
print("The test dataset contains {} labels, with the following distribution:\n
{}".format(np.shape(test_labels)[0], Counter(test_labels[:])))

uci_har_signals_train, uci_har_labels_train = randomize(train_signals,
np.array(train_labels))
uci_har_signals_test, uci_har_labels_test = randomize(test_signals,
np.array(test_labels))
```

```
The train dataset contains 7352 signals, each one of length 128 and 9 components
The test dataset contains 7352 signals, each one of length 128 and 9 components
The train dataset contains 7352 labels, with the following distribution:
 Counter({6: 1407, 5: 1374, 4: 1286, 1: 1226, 2: 1073, 3: 986})
The test dataset contains 2947 labels, with the following distribution:
 Counter({6: 537, 5: 532, 1: 496, 4: 491, 2: 471, 3: 420})
```

```
waveletname = 'rbio3.1'
X_train, Y_train = get_uci_har_features(uci_har_signals_train,
uci_har_labels_train, waveletname)
X_test, Y_test = get_uci_har_features(uci_har_signals_test, uci_har_labels_test,
waveletname)


models = batch_classify(X_train, Y_train, X_test, Y_test)
display_dict_models(models)

df_train = pd.read_csv("../input/mitbih_train.csv", header=None)
df_train = df_train.sample(frac=1)
df_test = pd.read_csv("../input/mitbih_test.csv", header=None)

Y = np.array(df_train[187].values).astype(np.int8)
X = np.array(df_train[list(range(187))].values)[..., np.newaxis]

Y_test = np.array(df_test[187].values).astype(np.int8)
X_test = np.array(df_test[list(range(187))].values)[..., np.newaxis]


def get_model():
    nclass = 5
    inp = Input(shape=(187, 1))
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(inp)
```

```
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = GlobalMaxPool1D()(img_1)
    img_1 = Dropout(rate=0.2)(img_1)

    dense_1 = Dense(64, activation=activations.relu, name="dense_1")(img_1)
    dense_1 = Dense(64, activation=activations.relu, name="dense_2")(dense_1)
    dense_1 = Dense(nclass, activation=activations.softmax,
name="dense_3_mitbih")(dense_1)

    model = models.Model(inputs=inp, outputs=dense_1)
    opt = optimizers.Adam(0.001)

    model.compile(optimizer=opt, loss=losses.sparse_categorical_crossentropy,
metrics=['acc'])
    model.summary()
    return model

model = get_model()
file_path = "baseline_cnn_mitbih.h5"
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1,
save_best_only=True, mode='max')
early = EarlyStopping(monitor="val_acc", mode="max", patience=5, verbose=1)
redonplat = ReduceLROnPlateau(monitor="val_acc", mode="max", patience=3,
verbose=2)
```

```
callbacks_list = [checkpoint, early, redonplat]  # early

model.fit(X, Y, epochs=1000, verbose=2, callbacks=callbacks_list,
validation_split=0.1)
model.load_weights(file_path)

pred_test = model.predict(X_test)
pred_test = np.argmax(pred_test, axis=-1)

f1 = f1_score(Y_test, pred_test, average="macro")

print("Test f1 score : %s "% f1)

acc = accuracy_score(Y_test, pred_test)

print("Test accuracy score : %s "% acc)
waveletname = 'rbio3.1'
X_train, Y_train = get_uci_har_features(uci_har_signals_train,
uci_har_labels_train, waveletname)
X_test, Y_test = get_uci_har_features(uci_har_signals_test, uci_har_labels_test,
waveletname)


models = batch_classify(X_train, Y_train, X_test, Y_test)
display_dict_models(models)

df_train = pd.read_csv("../input/mitbih_train.csv", header=None)
df_train = df_train.sample(frac=1)
df_test = pd.read_csv("../input/mitbih_test.csv", header=None)

Y = np.array(df_train[187].values).astype(np.int8)
X = np.array(df_train[list(range(187))].values)[..., np.newaxis]

Y_test = np.array(df_test[187].values).astype(np.int8)
X_test = np.array(df_test[list(range(187))].values)[..., np.newaxis]


def get_model():
    nclass = 5
    inp = Input(shape=(187, 1))
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(inp)
```

```
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = GlobalMaxPool1D()(img_1)
    img_1 = Dropout(rate=0.2)(img_1)

    dense_1 = Dense(64, activation=activations.relu, name="dense_1")(img_1)
    dense_1 = Dense(64, activation=activations.relu, name="dense_2")(dense_1)
    dense_1 = Dense(nclass, activation=activations.softmax,
name="dense_3_mitbih")(dense_1)

    model = models.Model(inputs=inp, outputs=dense_1)
    opt = optimizers.Adam(0.001)

    model.compile(optimizer=opt, loss=losses.sparse_categorical_crossentropy,
metrics=['acc'])
    model.summary()
    return model

model = get_model()
file_path = "baseline_cnn_mitbih.h5"
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1,
save_best_only=True, mode='max')
early = EarlyStopping(monitor="val_acc", mode="max", patience=5, verbose=1)
redonplat = ReduceLROnPlateau(monitor="val_acc", mode="max", patience=3,
verbose=2)
```

```
callbacks_list = [checkpoint, early, redonplat]  # early

model.fit(X, Y, epochs=1000, verbose=2, callbacks=callbacks_list,
validation_split=0.1)
model.load_weights(file_path)

pred_test = model.predict(X_test)
pred_test = np.argmax(pred_test, axis=-1)

f1 = f1_score(Y_test, pred_test, average="macro")

print("Test f1 score : %s "% f1)

acc = accuracy_score(Y_test, pred_test)

print("Test accuracy score : %s "% acc)


import pandas as pd
import numpy as np

from keras import optimizers, losses, activations, models
from keras.callbacks import ModelCheckpoint, EarlyStopping,
LearningRateScheduler, ReduceLROnPlateau
from keras.layers import Dense, Input, Dropout, Convolution1D, MaxPool1D,
GlobalMaxPool1D, GlobalAveragePooling1D, \
    concatenate
from sklearn.metrics import f1_score, accuracy_score


df_train = pd.read_csv("../input/mitbih_train.csv", header=None)
df_train = df_train.sample(frac=1)
df_test = pd.read_csv("../input/mitbih_test.csv", header=None)

Y = np.array(df_train[187].values).astype(np.int8)
X = np.array(df_train[list(range(187))].values)[..., np.newaxis]

Y_test = np.array(df_test[187].values).astype(np.int8)
X_test = np.array(df_test[list(range(187))].values)[..., np.newaxis]


def get_model():
    nclass = 5
```

```
    inp = Input(shape=(187, 1))
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(inp)
    img_1 = Convolution1D(16, kernel_size=5, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(32, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = MaxPool1D(pool_size=2)(img_1)
    img_1 = Dropout(rate=0.1)(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = GlobalMaxPool1D()(img_1)
    img_1 = Dropout(rate=0.2)(img_1)

    dense_1 = Dense(64, activation=activations.relu, name="dense_1")(img_1)
    dense_1 = Dense(64, activation=activations.relu, name="dense_2")(dense_1)
    dense_1 = Dense(nclass, activation=activations.softmax,
name="dense_3_mitbih")(dense_1)

    model = models.Model(inputs=inp, outputs=dense_1)
    opt = optimizers.Adam(0.001)

    model.compile(optimizer=opt, loss=losses.sparse_categorical_crossentropy,
metrics=['acc'])
    model.summary()
    return model

model = get_model()
file_path = "baseline_cnn_mitbih.h5"
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1,
save_best_only=True, mode='max')
```

```
early = EarlyStopping(monitor="val_acc", mode="max", patience=5, verbose=1)
redonplat = ReduceLROnPlateau(monitor="val_acc", mode="max", patience=3,
verbose=2)
callbacks_list = [checkpoint, early, redonplat]  # early

model.fit(X, Y, epochs=1000, verbose=2, callbacks=callbacks_list,
validation_split=0.1)
model.load_weights(file_path)

pred_test = model.predict(X_test)
pred_test = np.argmax(pred_test, axis=-1)

f1 = f1_score(Y_test, pred_test, average="macro")

print("Test f1 score : %s "% f1)

acc = accuracy_score(Y_test, pred_test)

print("Test accuracy score : %s "% acc)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = GlobalMaxPool1D()(img_1)
    img_1 = Dropout(rate=0.2)(img_1)

    dense_1 = Dense(64, activation=activations.relu, name="dense_1")(img_1)
    dense_1 = Dense(64, activation=activations.relu, name="dense_2")(dense_1)
    dense_1 = Dense(nclass, activation=activations.softmax,
name="dense_3_mitbih")(dense_1)

    model = models.Model(inputs=inp, outputs=dense_1)
    opt = optimizers.Adam(0.001)

    model.compile(optimizer=opt, loss=losses.sparse_categorical_crossentropy,
metrics=['acc'])
    model.summary()
    return model

model = get_model()
file_path = "baseline_cnn_mitbih.h5"
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1,
save_best_only=True, mode='max')
```

```
early = EarlyStopping(monitor="val_acc", mode="max", patience=5, verbose=1)
redonplat = ReduceLROnPlateau(monitor="val_acc", mode="max", patience=3,
verbose=2)
callbacks_list = [checkpoint, early, redonplat]  # early

model.fit(X, Y, epochs=1000, verbose=2, callbacks=callbacks_list,
validation_split=0.1)
model.load_weights(file_path)

pred_test = model.predict(X_test)
pred_test = np.argmax(pred_test, axis=-1)

f1 = f1_score(Y_test, pred_test, average="macro")

print("Test f1 score : %s "% f1)

acc = accuracy_score(Y_test, pred_test)

print("Test accuracy score : %s "% acc)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = Convolution1D(256, kernel_size=3, activation=activations.relu,
padding="valid")(img_1)
    img_1 = GlobalMaxPool1D()(img_1)
    img_1 = Dropout(rate=0.2)(img_1)

    dense_1 = Dense(64, activation=activations.relu, name="dense_1")(img_1)
    dense_1 = Dense(64, activation=activations.relu, name="dense_2")(dense_1)
    dense_1 = Dense(nclass, activation=activations.softmax,
name="dense_3_mitbih")(dense_1)

    model = models.Model(inputs=inp, outputs=dense_1)
    opt = optimizers.Adam(0.001)

    model.compile(optimizer=opt, loss=losses.sparse_categorical_crossentropy,
metrics=['acc'])
    model.summary()
    return model

model = get_model()
file_path = "baseline_cnn_mitbih.h5"
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1,
save_best_only=True, mode='max')
```

```
early = EarlyStopping(monitor="val_acc", mode="max", patience=5, verbose=1)
redonplat = ReduceLROnPlateau(monitor="val_acc", mode="max", patience=3,
verbose=2)
callbacks_list = [checkpoint, early, redonplat]  # early

model.fit(X, Y, epochs=1000, verbose=2, callbacks=callbacks_list,
validation_split=0.1)
model.load_weights(file_path)

pred_test = model.predict(X_test)
pred_test = np.argmax(pred_test, axis=-1)

f1 = f1_score(Y_test, pred_test, average="macro")

print("Test f1 score : %s "% f1)

acc = accuracy_score(Y_test, pred_test)

print("Test accuracy score : %s "% acc)
```